UNIVERSIDAD CARLOS III DE MADRID

PROYECTO FIN DE GRADO

DESARROLLO DE LIBRERÍA DE ESTIMACIÓN DE DISTANCIAS PARA APLICACIÓN MÓVIL. APLICACIÓN A VEHÍCULOS.

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Autor: Carmen Gandía Fernández

Tutor: Fernando García Fernández

Leganés, 24 de Septiembre de 2014

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1 Motivación

Con motivo de la creciente importancia que están tomando los *smartphones* en la vida cotidiana se está intentado mejorar muchas de sus funcionalidades.

Este trabajo pretende contribuir a la mejora en la ayuda en la conducción, en concreto en la detección y seguimiento de automóviles, para cualquier *smartphone* con sistema operativo Android. Se basa en el desarrollo de una biblioteca para el cálculo de distancias entre el vehículo y la cámara del Smartphone. Además esta biblioteca facilita los parámetros intrínsecos de la cámara y gracias al acelerómetro se conoce la rotación y aceleración del dispositivo en cada momento.

Por tanto, la biblioteca está desarrollada con el propósito de facilitar la implementación de nuevas aplicaciones que sirvan como soporte en la navegación; por ejemplo, calcular la distancia y la dirección de los coches para avisar de una posible colisión.

1.2 Entorno Socio-Económico

Según la DGT (Dirección General d Tráfico), solamente en España durante el año 2013 hubo 89.519 accidentes con víctimas, tanto en zona urbana como interurbana, de ellos 49.280 fueron accidentes entre vehículos con un total de 621 fallecidos *[1].* Ese mismo año fueron 11.026 los atropellos a peatones en España, con un total de 349 fallecidos *[1].*

Tal y como se muestra en la siguiente tabla, la OMS (Organización Mundial de la Salud), prevé que para 2030 los traumatismos por accidentes de tráfico aumenten para pasar a ser la quinta causa principal de mortalidad *[2].*

TOTAL 2004

NO. DE ORDEN	PRINCIPALES CAUSAS	%
1	Enfermedad isquémica del corazón	12,2
2	Enfermedad cerebrovascular	9,7
3	Infecciones de las vías respiratorias inferiores	7,0
4	Enfermedad pulmonar obstructiva crónica	5,1
5	Enfermedades diarreicas	3,6
6	VIH/sida	3,5
7	Tuberculosis	2,5
8	Cánceres de la tráquea, los bronquios y el pulmón	2,3
9	Traumatismos por accidentes de tránsito	2,2
10	Prematuridad y bajo peso al nacer	2,0
11	Infecciones neonatales y otras*	1,9
12	Diabetes mellitus	1,9
13	Paludismo	1,7
14	Enfermedad cardiaca hipertensiva	1,7
15	Asfixia del nacimiento y traumatismo del nacimiento	1,5
16	Lesiones autoinfligidas	1,4
17	Cáncer del estómago	1,4
18	Cirrosis del hígado	1,3
19	Nefritis y nefrosis	1,3
20	Cáncer colorectal	1,1

TOTAL 2030

NO. DE ORDEN	PRINCIPALES CAUSAS	%
1	Enfermedad isquémica del corazón	14,2
2	Enfermedad cerebrovascular	12,1
3	Enfermedad pulmonar obstructiva crónica	8,6
4	Infecciones de las vías respiratorias inferiores	3,8
5	Traumatismos por accidentes de tránsito	3,6
6	Cánceres de la tráquea, los bronquios y el pulmón	3,4
7	Diabetes mellitus	3,3
8	Enfermedad cardiaca hipertensiva	2,1
9	Cáncer del estómago	1,9
10	VIH/sida	1,8
11	Nefritis y nefrosis	1,6
12	Lesiones autoinfligidas	1,5
13	Cáncer del hígado	1,4
14	Cáncer colorectal	1,4
15	Cáncer del esófago	1,3
16	Violencia	1,2
17	Alzheimer y otras demencias	1,2
18	Cirrosis del hígado	1,2
19	Cáncer de mama	1,1
20	Tuberculosis	1,0

Ilustración 1. Principales causas de mortalidad, datos comprendidos entre 2004 y 2030.

Este trabajo está enfocado a facilitar una conducción segura tanto como para los pasajeros, los cuales según las estadísticas tienen más riesgo en colisiones con otros vehículos, como para proteger a los viandantes pues son el elemento más vulnerable. Según la OMS los peatones junto con los ciclistas y motocicletas representan más de la mitad de las víctimas mortales en todo el mundo.

La tecnología es una pieza importante en el crecimiento de la seguridad vial, un claro ejemplo de ello es la bajada de los accidentes con víctimas mortales a partir de los años 90, pese aumentar el número de vehículos en circulación *[1].* Otro ejemplo de que la tecnología cumple con su papel es que más del 90% de las víctimas mortales en accidentes de tráfico corresponde a países con ingresos bajos y menos desarrollados *[2].*

Aunque la verdadera tragedia de los accidentes de tráfico son las pérdidas de vidas humanas, también es importante destacar las pérdidas económicas. Según la FITSA (Fundación Instituto Tecnológico para la Seguridad del Automóvil), el coste social asociado a las víctimas de siniestros de circulación representa aproximadamente el 2% del todo el producto interior bruto, que equivale a un tercio de la riqueza que genera todo el sector de automoción en España, uno de los más importantes del país *[3].*

En los últimos diez años los accidentes de tráfico han representado para la sociedad española un coste comprendido entre 105.000 y 144.000 millones de euros. Estas cifras recogen todo tipo de costes, tanto administrativos como materiales y médicos, así como costes de pérdidas en la productividad y evidentemente costes humanos. Todos ellos representan unas pérdidas que ninguna sociedad puede hacer frente.

Con esta biblioteca se abre paso a un amplio abanico de posibilidades para mejorar la ayuda y la seguridad en la conducción, con la comodidad de usar un dispositivo que siempre llevamos encima, nuestro *smartphone*.

1.3 Estructura del Documento

Para facilitar la lectura del documento se ha divido en los siguientes apartados:

- **Introducción:** En este primer apartado se exponen las motivaciones para llevar a cabo el proyecto, el entorno socio económico que lo contextualiza y estructuración.
- **Estado del arte:** En esta sección se exponen los avances realizados en los sistemas de reconocimiento del entorno en la conducción, primero de manera general para posteriormente centrarse en dichos sistemas aplicados a dispositivos Android.
- **Descripción general:** En el siguiente apartado se explican los objetivos finales del proyecto y los recursos utilizados para llevarlos a cabo, tanto sensores (hardware) como el entorno en el que se desarrollan (software).
- **Algoritmo:** Este apartado recoge todos los modelos y metodologías de cálculo que han sido necesarios para crear la clase la biblioteca de trabajo.
- **Pruebas de Evaluación y Resultados:** En este capítulo se muestran todas las pruebas y sus correspondientes análisis, para dar consistencia a los cálculos que se han llevado a cabo.
- **Conclusión y Líneas Futuras**: En esta sección se recogen las conclusiones finales, resumiendo conceptos determinantes y haciendo una reflexión sobre las posibles aplicaciones futuras de este trabajo.

2. ESTADO DEL ARTE

Actualmente los sistemas de seguridad en los vehículos se pueden dividir en dos tipologías:

- **Seguridad pasiva:** estos sistemas se encargan de minimizar los daños en caso de accidente. Su eficacia está más que comprobada puesto que sistemas como el cinturón de seguridad ya son obligatorios en todos los vehículos y otros como el airbag se están implementando en la mayoría de ellos.

- **Seguridad activa**: este grupo de sistemas son aquellos que garantizan la seguridad y previenen los accidentes de tráfico, como por ejemplo el sistema de frenado, siendo el más destacado el ABS, o los sistemas de dirección, suspensión o control de la estabilidad.

Como hemos visto estos elementos contribuyen a mejorar la seguridad y eficacia del vehículo y muchos de ellos son obligatorios *[4]*. Actualmente las nuevas tecnologías están abriéndose paso en los sistemas de reconocimiento del entorno para anticipar y evitar los accidentes. Este proyecto se engloba dentro de esta última tipología de la que hablaremos más adelante.

2.1 Sistemas de reconocimiento del entorno

Diferentes estudios en todo el mundo demuestran que la principal causa de los accidentes de tráfico es el factor humano, cerca del 80% de los accidentes se deben a una distracción, las prisas, el cansancio o el consumo de alcohol *[5]*. Por lo que parece lógico desarrollar nuevos sistemas que ayuden a la corrección y anticipación de errores del conductor. Los siguientes puntos recogen los sistemas más importantes en la actualidad:

2.1.1 Detección de sueño o falta de atención al volante

También llamado detección de fatiga tiene como finalidad indicar si el conductor está en las condiciones apropiadas para seguir conduciendo. Normalmente es un sistema electrónico integrado en el volante que analiza varias veces por minuto las correcciones en la dirección, si el procesador cuenta con menos correcciones por minuto de lo que se considera normal el sistema da la señal de alerta interpretando que el conductor está distraído, fatigado o incluso durmiéndose al volante.

Este sistema está integrado en el Volkswagen Passat, pero también hay otros en fase más experimental basados en el reconocimiento facial, con una pequeña cámara, que detectan los parpadeos del conductor y dan la alerta si la frecuencia de parpadeo está fuera de los intervalos de normalidad *[6].*

Ilustración 2. Alarma de detección de sueño integrado en el Volkswagen Passat.

2.1.2 Detección de cambio de carril involuntario

También llamado asistente de mantenimiento en carril funciona mediante un microprocesador que está permanentemente asegurando que el coche permanezca entre las dos lineal del carril. Hay dos tipos de sistemas, el primero utiliza dos sensores colocados en el exterior a cada lado del vehículo mientras que otros fabricantes optan por una única cámara situada en lo alto del parabrisas.

Ambos sistemas dan el aviso mediante vibraciones en el volante o señales sonoras que el vehículo está saliendo del carril sin activar el intermitente. Algunos fabricantes van un paso más y corrigen ligeramente la dirección aunque desisten cuando el conductor ejerce fuerza para contrarrestar dicho movimiento. Modelos como el Citröen C4 ya lo tienen incorporado aunque solo se activa a velocidades superiores de 50 o 60 Km/h *[7].*

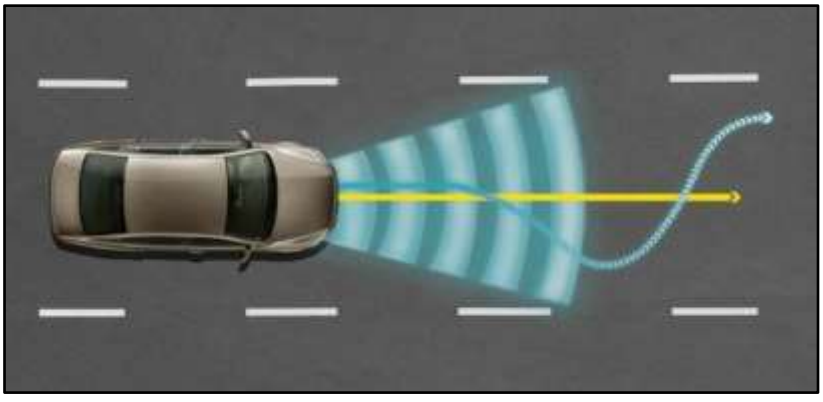


Ilustración 3. Detección de cambio de carril involuntario.

2.1.3 Advertencia de ángulo muerto en el espejo retrovisor

Es un sistema de equipamiento opcional en coches como el Audi A6 y consiste en vigilar el ángulo muerto del coche mediante radares colocados en las esquinas de los paragolpes traseros, en el lateral del coche o en el espejo retrovisor. Al detectar cualquier vehículo el sistema avisa al conductor mediante un led o triangulo de advertencia colocado en el espejo retrovisor, tal y como se muestra en la imagen *[8].*

Ilustración 4. Alarma de detección de obstáculo en el ángulo muerto, integrado en un Audi A6.

2.1.4 Reconocimiento de señales de velocidad y de adelantamiento

Este sistema también empieza a estar disponible en coches de gama media y precio asequible, como por ejemplo en el Opel Astra. El sistema está basado en una cámara de alta resolución y gran angular, colocada en la parte alta del parabrisas y centrada delante del espejo retrovisor, vigilando los márgenes de la calzada y reconociendo las señales circulares de velocidad máxima, prohibido adelantar, fin de velocidad máxima y fin de prohibido adelantar.

En la pantalla digital del cuadro de instrumentos se muestra de manera permanente la velocidad límite en el tramo en el que nos encontremos, y se va actualizando en tiempo real según cambien las señales *[9].*

Ilustración 5. Indicador de señales, integrado en el panel de control de un Opel Astra.

2.1.5 Advertencia de sentido contrario

Este sistema no es tan habitual, y hay que buscarlo en coches de alta gama como BMW. Aunque parezca extraño, cada año se dan muchos casos de conductores que circulan en sentido contrario a la marcha, en autovías y autopistas (en Alemania por ejemplo se dan unos 1.800 casos al año). Así que este sistema puede resultar más útil de lo que cabría pensar. Utiliza el sistema de GPS del coche para identificar si el conductor está a punto de incorporarse a una carretera en sentido contrario, y avisarle de ello con una alarma sonora y visual en la pantalla del navegador. También se está contemplando la posibilidad de complementar el sistema con la cámara de reconocimiento de señales.

Además de la alarma para el propio conductor, manda un aviso a los coches que estén cerca (en un radio de 600 m) que estén provistos de un sistema de comunicación y también manda un aviso a una central de tráfico, que pueda notificar el riesgo a todos los conductores de la carretera, mediante sistema de radio o mediante paneles digitales de información (lógicamente esto último requiere de cierta infraestructura) *[7].*

Ilustración 6. Indicador de sentido contrario integrado en el panel de control.

2.1.6 Control de velocidad adaptativo (con sistema de distancia de seguridad)

Este sistema es la evolución del control de velocidad permitiendo mantener la distancia óptima de seguridad con el vehículo que nos precede. Un radar colocado en el paragolpes delantero mide la distancia entre los vehículos y de acuerdo a la velocidad que llevemos el microprocesador calcula cual debería ser la distancia de seguridad y velocidad recomendables.

Si no hay suficiente distancia, entonces el sistema actúa sobre el acelerador reduciendo la velocidad. En el caso de que no resulte efectiva esta deceleración, el sistema puede actuar también sobre el freno para reducir la velocidad de forma más rápida y efectiva. Una vez que se ha conseguido una buena distancia de seguridad el coche vuelve a acelerar para recuperar la velocidad programada. Este sistema ya está disponible en coches como en el Toyota Prius.

Los sistemas más avanzados aprovechan también los sensores del ABS y del control de estabilidad para conocer el estado de adherencia del pavimento, saber por ejemplo si está mojado, y así adaptar la velocidad automáticamente *[10].*

Ilustración 7. Simulación de la distancia de seguridad de los vehículos con control adaptativo.

2.1.7 Reconocimiento de objetos

Este sistema pretende evitar sobre todo atropellos a peatones y ciclistas ya que son los elementos más vulnerables de la carretera. El primer sistema, llamado *ContiGuard*, funciona mediante dos cámaras de alta resolución que distinguen entre peatones, ciclistas, coches y otros objetos *[11]*. El sistema mide la distancia hasta ellos y predice su trayectoria, determinando si existe riesgo de accidente. De esta forma el conductor es advertido del peligro e incluso es posible que se accione el freno de forma automática.

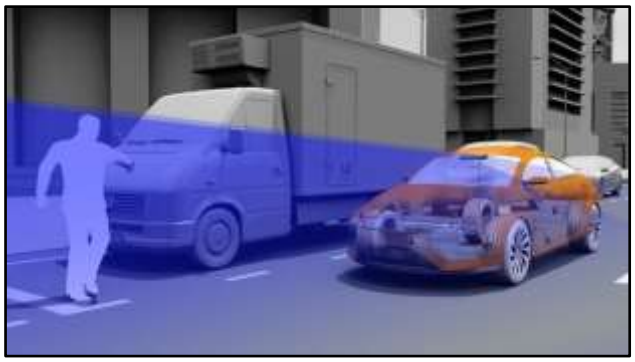


Ilustración 8. Simulación del sistema ContiGuard.

Otro sistema de reconocimiento de objetos, aún en fase experimental, es el llamado *Amulett Car2X*. Éste funciona por ondas de radio por lo que los peatones o ciclistas deben ir equipados con un pequeño transponedor que incluso no necesitaría alimentación eléctrica *[12].*

El coche debe llevar un emisor-receptor que recibe la señal de respuesta de los transpondedores. Lo bueno de este sistema es que los peatones podrían estar ocultos (por ejemplo niños detrás de un vehículo estacionado) pero el microprocesador del coche sabe que están ahí y advierte de ello al conductor.

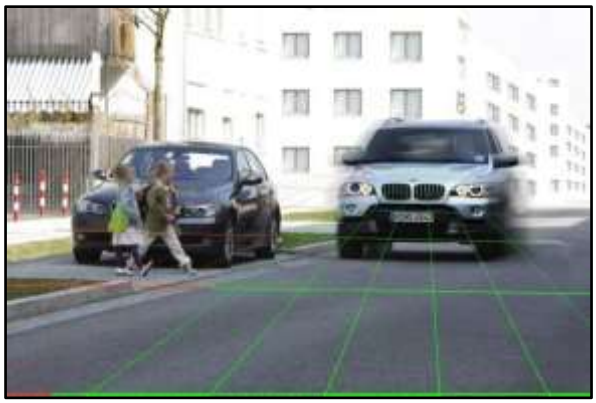


Ilustración 9. Simulación del sistema Amulett Car2X.

2.1.8 Sistemas de anticolisión (frenado automático)

Este sistema pretende reaccionar antes que el conductor, accionando los frenos automáticamente al detectar un obstáculo delante del vehículo, reduciendo al velocidad o incluso parando el coche para evitar el accidente.

Hay varios tipos de sistemas, un ejemplo, es el *City Safety* de Volvo el cual opera a velocidades bajas ya que está pensado para ciudades o atascos de tráfico, evitando accidentes por despiste *[13]*.

Ilustración 10. Simulación del sistema City Safety de Volvo.

Hay otros tipos, aún experimentales, como la anticipación a curvas de BMW que con ayuda del GPS se anticipa a las curvas y si la velocidad es superior a la óptima frena para reducir la velocidad y si fuera necesario también reduciría marchas en la caja de cambios.

Ilustración 11. Anuncio de presentación del sistema de anticipación a curvas de BMW.

2.1.9 Cámaras de visión

Un ejemplo es la cámara de visión para marcha atrás implementada y habitual en bastantes coches como el Toyota Auris HSD. Se basa en una cámara colocada en la puerta del maletero, encima de la matrícula que permite ver las imágenes en la pantalla a color del sistema de navegación de la consola central.

Esta cámara permite ver cosas pequeñas que podrían quedar ocultas de la visión de los espejos retrovisores exteriores y por tanto de la visión del conductor.

La evolución de esto ya existe también en coches BMW. Las cámaras de visión lateral, colocadas en las esquinas del paragolpes delantero permiten ver hacia la derecha y hacia la izquierda en cruces en los que el conductor no tiene suficiente visibilidad *[7].*

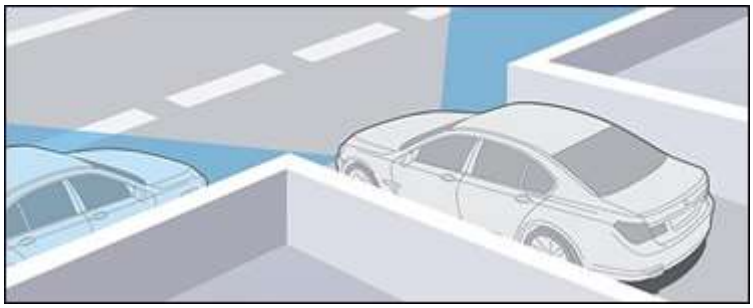


Ilustración 12. Simulación del sistema de cámaras de BMW.

2.1.10 Sistemas integrados en el coche de Google

El automóvil de Google es un proyecto de la compañía que consiste en el desarrollo de la tecnología necesaria para crear coches sin conductor, vehículos autónomos. Actualmente el líder del proyecto es el ingeniero alemán de Google Sebastian Thrun director del *Stanford Artificial Intelligence Laboratory* y coinventor de Google *Street View*.

El equipo de Thrun en *Stanford* creó el vehículo robótico Stanley, que fue el ganador del DARPA Grand Challenge en 2005, otorgado por el Departamento de Defensa de los Estados Unidos y dotado con un premio de 2 millones de dólares *[14].*

Este coche es capaz de conducir de forma autónoma tanto por ciudad como por carretera, detectando otros vehículos, señales de tráfico, peatones…etc.

Ilustración 13. Vehículo Stanley, ganador de la DARPA Grand Challenge en 2005.

El Stanley está construido sobre el Volkswagen Touareg estándar Europeo. Se eligió este modelo por su sistema de control *"drive by wire*" el cual consiste en sustituir los actuadores mecánicos del vehículo por actuadores electro-mecánicos o simplemente eléctricos.

Para poder navegar, el Stanley monta cinco sensores LIDAR en el techo y un sensor capaz de recrear el entorno en 3D al igual que el Radar, pero a diferencia de éste se consigue mediante luz reflejada y no ondas sonoras. Tienen un alcance de hasta 60 metros y 360 º, lo cual es realmente importante.

Estos sensores complementan a la navegación por GPS que es fundamental. También incorpora una serie de sensores como giróscopos y acelerómetros que monitorizan la orientación del vehículo y sirven para complementar al GPS.

Además, una cámara proporciona datos adicionales de navegación hasta 8 metros de distancia por medio de visión por computador la cual complementa a los anteriores sensores mencionados.

Por último, el Stanley incorpora unos sensores en las ruedas los cuales entran en juego en caso de que se pierda la señal de navegación por GPS. Estos sensores actúan como calculadores de distancia recorrida desde el momento en que se pierde la señal para que puedan realizar el seguimiento aunque ésta se pierda. Todos los datos de los sensores son procesados por un computador equipado con seis Intel Pentium M a 1.6 GHz de bajo consumo, trabajando en diferentes versiones de Sistema Operativo Linux.

A día de hoy google ha dado un paso hacia delante y ha decidido que es hora de dejar de montar sus tecnologías en otros modelos y crear su propio prototipo. Se trata de un pequeño biplaza de formas redondeadas y no tiene volante, acelerador ni frenos.

Ilustración 14. Prototipo de vehículo autónomo de Google.

A diferencia de los Prius y Lexus con los que google ha estado trabajando, este prototipo no necesita más interacción humana que la de definir un destino sobre Google *Maps.*

El vehículo cuenta con sensores que analizan cientos de objetos simultáneamente en 360 º y a 180 metros de distancia *[15]*.

Google fabricará un centenar de estos prototipos a los que dotará de controles convencionales para dar mayor seguridad a sus pilotos. Las pruebas empezarán en California y puede que se extiendan durante años. La estrategia de Google no es poner este coche a la venta, si no perfeccionar el prototipo para poder vender la tecnología a los fabricantes de automóviles.

2.2 Sistemas de ayuda en la conducción para dispositivos Android

Observando todos los sistemas anteriormente vistos, es obvio que la tecnología ya lleva perfeccionando y desarrollando nuevos sistemas de ayuda en la navegación. El inconveniente es que son sistemas de equipamiento opcional y muchos de ellos solo disponibles en vehículos de alta gama que no están al alcance de la mayoría. Es en este punto donde parece lógico desarrollar una aplicación que incluso pueda contener todas las anteriores, sin necesidad de equipar el vehículo, únicamente con un *smartphone* que se podría incorporar en cualquier vehículo sin necesidad de equipamiento.

Por este motivo se ha elegido para el proyecto el sistema operativo Android, porque está disponible en terminales de alta, media y baja gama, y con una comprobada fiabilidad. Es razonable que se esté intentado unificar todos los sistemas de reconocimiento del entorno para la ayuda en la conducción, en un único dispositivo que además llevamos siempre encima, estos son algunos de los ejemplos que ya se han diseñado con dicho propósito y que están disponibles en la tienda de Android.

2.2.1 Waze

Es la aplicación de conducción más famosa y con más éxito de todos los sistemas operativos para móviles. Tiene 40 millones de usuarios e informa del estado del tráfico en todo el mundo.

Gracias al GPS del teléfono podemos compartir nuestra ubicación en una red social de conductores donde el sistema informa de retenciones, tráfico lento, accidentes o vías cortadas. Con el móvil colocado a modo de navegador GPS, se puede ver en la pantalla un mapa de la ruta con los atascos según vayan ocurriendo. Es gratuita *[16].*

Ilustración 15. Aplicación Waze.

2.2.2 iOnRoad Augmented Driving

Este programa utiliza los diversos sensores del teléfono y la cámara, para avisarnos de posibles colisiones con otros vehículos y otros peligros**.**

El móvil se coloca con un soporte en el salpicadero, de modo que permita enfocar la cámara al frente a través de la luna delantera. Así transmite la conducción en tiempo real y hace que aparezcan en la pantalla diferentes datos, como la distancia de seguridad recomendada y la que realmente guardamos, o si excedemos la velocidad o nos salimos del carril. Cuesta 3,99 euros *[17]***.**

Ilustración 16. Aplicación iOnRoad Augmented Driving

2.2.3 Drivea

Similar a la anterior aplicación, este software está más centrado en calibrar los riesgos que se puedan producir y en advertir de si hay posibilidad de colisión o nos salimos de carril o encaramos una curva a excesiva velocidad. También usa los sensores del teléfono para calcular los peligros. Se sitúa con un soporte sobre el salpicadero y frente a la luna y es gratuita *[18].*

Ilustración 17. Aplicación Drivea.

2.2.4 Car Home Ultra

Este sistema es un navegador GPS enriquecido; estudia el consumo energético, la meteorología en el trayecto o la altitud a la que nos encontramos. También avisa si se supera la velocidad recomendada y tiene la opción de contestar llamadas de forma automática que indican que estamos en plena conducción. Es una aplicación gratuita *[19]*.

Ilustración 18. Aplicación Car Home Ultra

2.2.5 DGT

Es la aplicación oficial de la Dirección General de Tráfico. Permite ver las rutas elegidas por carretera para conocer el estado de las carreteras y si hay o no algún tipo de incidencia. Dispone de avisos con sonido para incidencias graves con acceso directo a urgencia 112, e incluso la posibilidad de ver las cámaras de tráfico y ver la ubicación de los radares de carretera *[20].*

Ilustración 19. Aplicación DGT.

3. DESCRIPCIÓN GENERAL

3.1 Propósito

En un principio el propósito del trabajo era desarrollar una aplicación que permitiese calcular la distancia hacia un peatón-coche a través de la cámara de cualquier dispositivo Android. Esta aplicación sería la continuación de un reconocimiento de coches, pero como esta primera parte no está perfeccionada y detecta los patrones de forma muy intermitente, continuar con la aplicación no cumple con el objetivo principal: darle una utilidad práctica a este trabajo. Por este motivo se vio que era más conveniente el desarrollo de una clase llamada *DistanceEstimation*, en la que se recogen todas las funciones de obtención de parámetros para tener una localización precisa del objeto de interés a través de nuestro dispositivo móvil.

3.2 Hardware

Para llevar a cabo dicho propósito es necesario obtener ciertos parámetros de interés propios de cada cámara: distancia focal, ángulo de visión y tamaño del sensor y de esta manera poder adaptar la aplicación a todo tipo cámaras. Además de los datos que nos proporciona el acelerómetro: gravedad, aceleración, pitch y roll; los cuales permitirán conocer la posición relativa del dispositivo con respecto al entorno.

Una de las complicaciones del proyecto es que todos estos dispositivos deben ser los que llevan integrados la mayoría de los *smartphones*, de esta manera se le aportará al usuario mayor comodidad a cambio de las limitaciones de hardware propias de dicho dispositivo.

3.2.1 Cámara

Es sabida la explosiva evolución de las cámaras en los dispositivos móviles en los últimos años. Llegando a desplazar a las cámaras compactas, los *smartphones* han adquirido una gran calidad que oscila entre los 8 y 12 megapíxeles pero que puede alcanzar hasta los 41 Mpx como es el caso del Nokia Lumia 1020 *[21], [22].* Por ejemplo, en la siguiente imagen se puede observar la evolución de la cámara del iPhone:

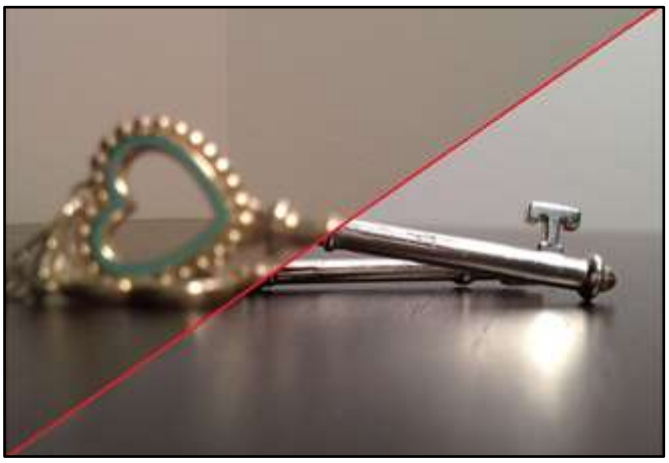


Ilustración 20. Evolución de las cámaras de iPhone.

La imagen izquierda corresponde a un iPhone original (lanzado en 2007), mientras que la de la derecha se corresponde con un iPhone 4S lanzado solamente 4 años después, en el 2011 *[23].* Es por tanto evidente el creciente potencial de estos dispositivos y parece natural intentar aprovechar dicho potencial con unos recursos, que aunque limitados, están en constante evolución.

3.2.2 Acelerómetro

Es un sensor que mide la aceleración relativa. Debido a su bajo coste (menos de un euro), está incluido en todos los *smartphones* y es el responsable de indicar si el dispositivo está en horizontal o en vertical, muy útil para facilitarnos el manejo. Esta es solo la más sencilla y obvia de sus funcionalidades, también es responsable de los sistemas anti-rotura que intentan minimizar los daños apagando el dispositivo al detectar una caída libre *[24].* Este pequeño y complejo sensor muestra el efecto de la aceleración de la gravedad en cada uno de los ejes, indicando la posición relativa del *smartphone* respecto al suelo, una utilidad que se ha aprovechado para corregir las deformaciones de perspectiva en la cámara habiendo más precisa la aplicación.

Ilustración 21. Acelerómetro.

3.3 Software: entorno de desarrollo

Una vez conocidos los sensores que son necesarios para desarrollar una aplicación de localización de vehículos, es necesario obtener la información que aportan dichos sensores, centrándonos en el software. Para este proyecto se ha utilizado el entorno de desarrollo de Eclipse con programación en Android y para el procesado de imágenes se usan las bibliotecas OpenCv.

3.3.1 Eclipse

Es el programa escogido por su gran éxito y por su versatilidad en el uso de herramientas externas a él, como la biblioteca OpenCv. En este caso, aunque admite una gran variedad de lenguajes de programación, se ha optado por usar el plugin de Android ADT (*Android Development Tool*), para hacer la aplicación compatible con los dispositivos en los que será usada.

Ilustración 22. Logotipo de Eclipse.

3.3.2 Android

Este sistema operativo de Google supera el 80% de la cuota de mercado a nivel mundial en *smartphones*, según la IDC (*International Data Corporation*) *[25]*. Este es el motivo principal por el que se ha elegido Android ya que sería compatible con la mayoría de dispositivos. Aunque cabe destacar que esto también puede suponer una complicación pues se debe adaptar y estandarizar a una gran cantidad de terminales distintos.

Ilustración 23. Logotipo de Android.

Otro de los motivos es que el software de Android es gratuito y posee muchas plataformas de apoyo (blogs, foros, etc), donde puedes encontrar desde la información más básica hasta cómo darle un uso más profesional *[26]*.

Su sencillez es otro punto importante, por ejemplo, mediante clases como *Camera* o *SensorManager* se puede acceder fácilmente a los sensores integrados en el *smartphone*, no solo para manejarlos sino para obtener información sobre ellos y hacerlos parte activa de la nueva aplicación.

3.3.3 OpenCv

Es una biblioteca proporcionada por Google de código abierto que se usan para el procesamiento de imágenes. Son un elemento crucial en cualquier aplicación basada en la localización, pues no se podría localizar el objeto de interés mediante patrones, ni hacer una pretratado si fuese necesario, ni crear una interfaz que facilite al usuario una detección rápida y segura durante la conducción.

Al estar enfocado a la programación en *smartphones* se ha usado la versión de OpenCv para Android basada en el lenguaje Java. Dicha librería no contiene todos los métodos que en su lenguaje nativo (C/C++), pero no solo son suficientes para esta tarea, sino que si estuvieran implementados métodos más complejos, el procesador del Smartphone no tendría la suficiente capacidad.

Ilustración 24. Logotipo de Android.

4. ALGORITMO

4.1 Parámetros Intrínsecos

Los parámetros intrínsecos son aquellos que definen la geometría interna y la óptica de la cámara, por lo que serán constantes en cada dispositivo, a menos que varíen las características o posiciones relativas entre la óptica y el sensor.

La cámara es uno de los pilares más importantes del trabajo por lo que se va a explicar detalladamente aquellos parámetros intrínsecos que han sido necesarios obtener para los posteriores cálculos.

4.1.1 Distancia focal

Es la distancia entre la óptica y el punto focal (donde se concentran todos los rayos). Este es un parámetro característico de cada cámara y al ser un mercado tan amplio y variado, Android incluye una clase llamada *Camera.Paramenters* que facilita este dato mediante la función *getFocalLength(), [27].* Posteriormente (*sección 4.1.3 Modelo PinHole*), se explicará la importancia y en que cálculos interviene dicho parámetro.

4.1.2 Tamaño de sensor

En el sensor se sitúan los sensores sensibles a la luz, por lo que su tamaño es proporcional a la resolución de la imagen capturada. Este parámetro no es proporcionado por Android directamente y además los fabricantes no suelen incluirlo en las hojas de características de los *smartphones*, por lo que debe ser calculado a través del ángulo de visión. En las siguientes imágenes se observa qué es el ángulo de visión, diferenciando el ángulo vertical del horizontal, que corresponderán con el alto y el ancho del sensor respectivamente:

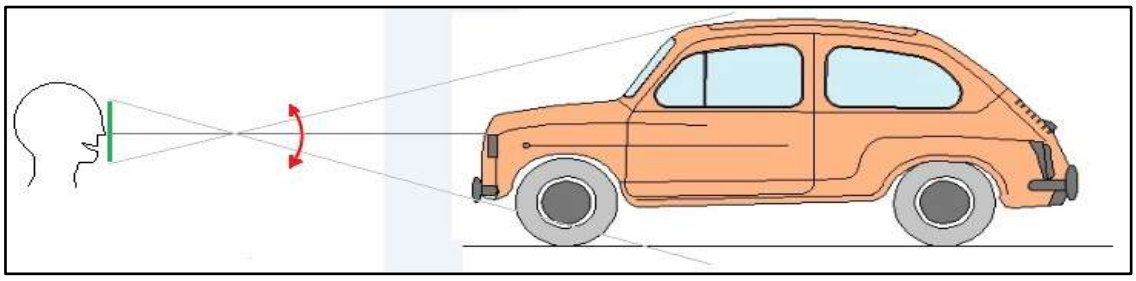


Ilustración 25. Ángulo de visión vertical.

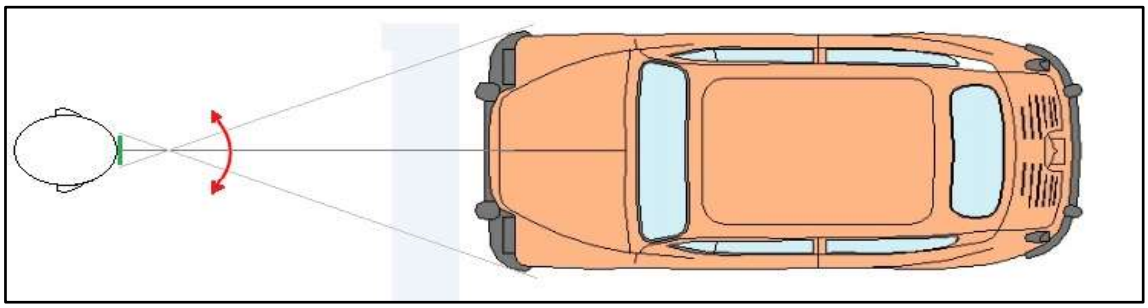


Ilustración 26. Ángulo de visión horizontal.

Si ampliamos la zona de interés se observa que mediante la trigonometría se puede relacionar el ángulo de visión con la distancia focal y el tamaño del sensor de la siguiente manera:

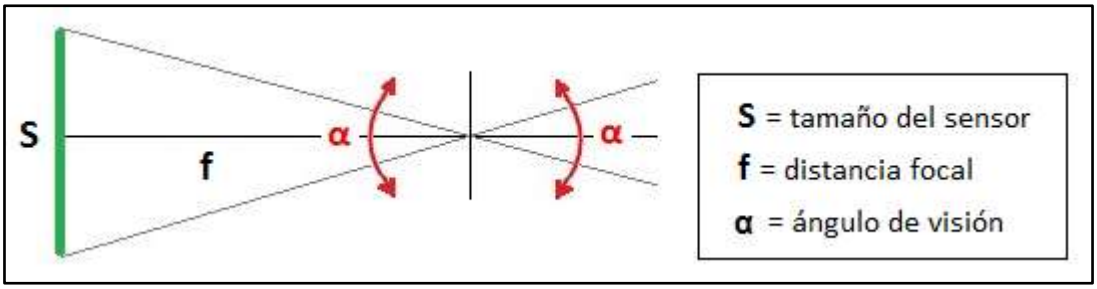


Ilustración 27. Esquema del ángulo de visión.

$$\tan\frac{\alpha}{2} = \frac{S/2}{f} \quad \rightarrow \quad S = 2 \cdot f \cdot \tan\frac{\alpha}{2}$$

Así se obtienen las dimensión del sensor tanto en vertical (el largo), como en horizontal (el ancho).

4.1.3 Punto central de enfoque de la cámara

Se trata del pixel de la imagen el cual no sufre refracción de la lente debido a que se encuentra en el punto central del sensor. En cámaras de alta gama suele estar indicado, pero en este caso Android no proporciona ninguna información. Lógicamente se ha elegido el punto central de la imagen puesto que suele coincidir o estar muy próximo a este.

De este modo se obtienen los parámetros intrínsecos de cualquier dispositivo Android, abarcando una gama muy alta de modelos de cámara haciendo posible la tan buscada compatibilidad.

4.2 Parámetros Extrínsecos

4.2.1 Acelerómetro: cálculo del *Pitch*

La otra base del trabajo consiste en posicionar el móvil con respecto al entorno, es decir, calcular la rotación en todos sus ejes para determinar la deformación que sufre la imagen y corregirla en el cálculo de distancias, lo cual se obtiene con el acelerómetro.

El acelerómetro es un dispositivo que mide la aceleración y las fuerzas inducidas por la gravedad, es decir, el movimiento y los giros. Para su manejo Android proporciona varias clases:

- ***Sensor:*** es una clase que representa un sensor y que se utiliza para obtener la lista de sensores disponibles en cada dispositivo *[28].* Aunque actualmente la mayoría de los *smartphones* tienen una amplia lista de sensores integrados el primer paso debe ser comprobar cuales están disponibles para optimizar cualquier aplicación.

- ***SensorManager*:** es una clase que permite acceder al sensor y siendo necesario indicar la tasa de lectura de datos al hacer el registro del acelerómetro *[29].*

- ***SensorEventListener:*** es una clase que sirve para el manejo de las actualizaciones del sensor, de esta manera se pueda hacer una toma continua de datos para corregir el error producido por los ángulos, en todo momento *[30].*

- ***SensorEvent:*** esta clase se usa para obtención de los datos del acelerómetro (entre otros mucho sensores), es decir, facilita la aceleración en cada uno de los tres ejes del dispositivo *[31].* Esta última clase es de especial interés debido a que es la que proporciona los datos para la orientación.

El propósito del acelerómetro es obtener los ángulos de rotación puesto que serán equivalentes a los ángulos de la deformación que puede sufrir una imagen. En las siguientes imágenes se enumeran y se explica que puede implicar dicha deformación en el cálculo de distancias. Para ello se usan los ángulos de Euler que son un conjunto de tres coordenadas angulares que sirven para especificar la orientación de un sistema de referencia móvil, respecto a otro sistema de referencia fijo, ambos de ejes ortogonales *[31].* De esta manera se puede especificar la posición de un sistema mediante tres ángulos: *Yaw, Pitch y Roll.*

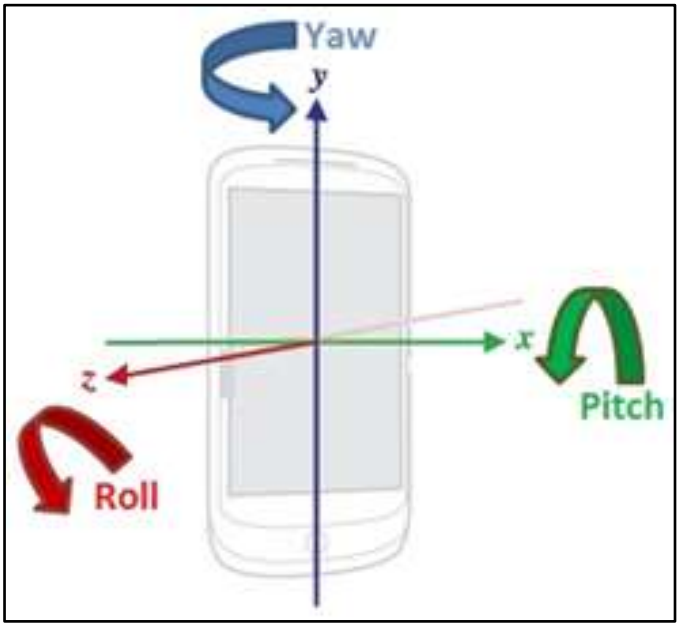


Ilustración 28. Representación de los Ángulos Euler en un Smartphone.

Los ejes del dispositivo en vertical están fijados como se indica en la *ilustración 28*, pero para optimizar la pantalla del *smartphone* todos los cálculos se han realizado en posición horizontal, con el consiguiente cambio de ejes:

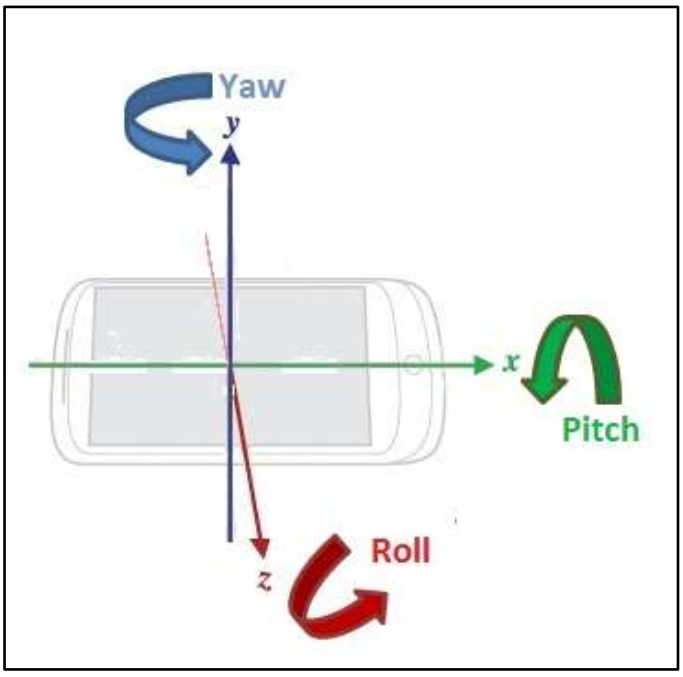


Ilustración 29. Representación de los Ángulos Euler en un Smartphone, posición horizontal.

En las siguientes figuras se muestra cómo afectan los ángulos de Euler en la captura de una imagen, centrando la atención en el punto de interés donde el programa de reconocimiento detectaría un vehículo. Este punto coincide con el suelo puesto que será el punto de referencia para el cálculo de la distancia hasta él.

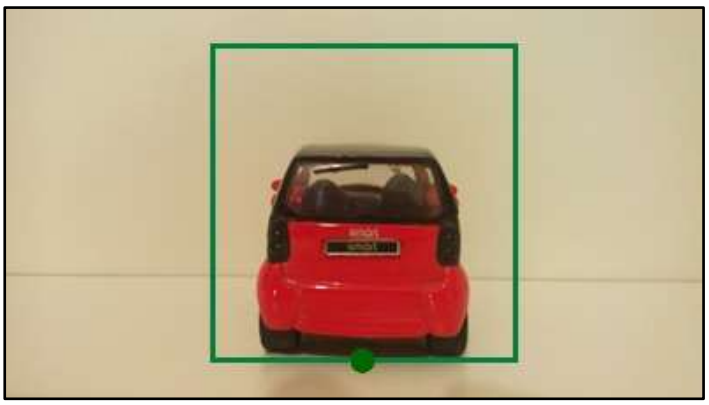


Ilustración 30. Imagen sin ángulos.

Ilustración 31. Yaw positivo.

Ilustración 32. Yaw negativo.

El punto de interés ha sufrido un desplazamiento horizontal. Su componente se desplaza hacia la derecha con un *Yaw* positivo, haciendo que el vehículo parezca más desplazado hacia este lado, y a la inversa con el negativo.

Ilustración 33. Pitch positivo.

Ilustración 34. Pitch negativo.

El punto de interés ha sufrido un desplazamiento puramente vertical. En el caso del pitch positivo el punto de interés se desplaza hacia arriba, mostrando un coche cuya perspectiva lo hace parecer más alejado. En el caso opuesto se pierde el punto de interés puesto que desaparece la visión del suelo y por tanto la referencia para el cálculo de las distancias.

Ilustración 35. Roll positivo.

Ilustración 36. Roll negativo.

El punto de interés ha sufrido un desplazamiento diagonal, en cada caso de manera opuesta. Además cuanto más alejado este el punto de interés del eje vertical central, mayor será su desplazamiento.

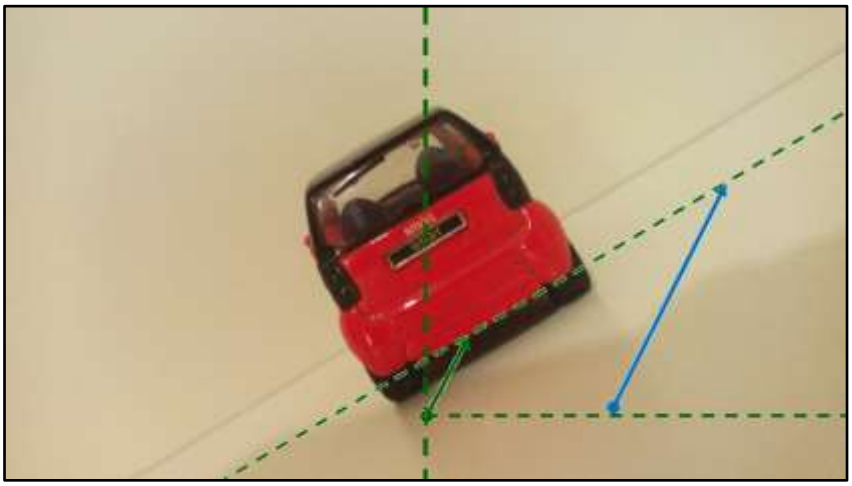


Ilustración 37. Influencia del Roll.

En la imagen se representa un vehículo cercano al eje vertical central, la flecha verde se asocia al desplazamiento que ha sufrido el punto de interés cuando aplicando el *Roll*. La flecha azul representa el desplazamiento sufrido si el vehículo se situase más alejado del eje central, siendo evidente que la deformación es mucho más pronunciada cuanto más alejado está el vehículo de dicho eje central.

Aunque como se ha visto, todos los ángulos afectan a las imágenes, para el cálculo de distancias básicamente se necesita el *Pitch*, por las siguientes razones:

- El *Pitch* es el ángulo de Euler que más deforma la profundidad de la imagen, su componente mayoritaria sufre el desplazamiento en el mismo eje que se usa para el cálculo de distancias hacia el vehículo, haciendo que se vea ampliamente modificada.
- El *Roll* no se calcula debido a que los patrones para el reconocimiento de coches no tienen en cuenta esta rotación, por lo que el dispositivo se debe colocar sin *Roll*, es decir, totalmente horizontal.
- El *Yaw*, aunque modifica las distancias hacia ambos lados, no se puede calcular porque el acelerómetro no da información sobre el plano perpendicular a la gravedad.

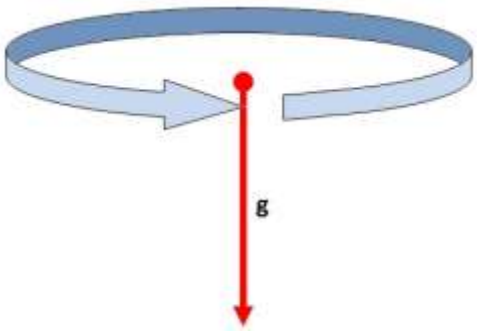


Ilustración 38. Yaw y gravedad.

En la imagen se observa que como el *Yaw* se produce en el mismo eje que el de la gravedad, aun rotando el dispositivo (representado por la flecha azul), la gravedad no se verá afectada.

Una vez conocidas las deformaciones y como afectan en el propósito, se va a explicar detalladamente como se calcula el *Pitch*:

Se parte de los datos proporcionado por el acelerómetro, sacados de la clase *SensorEvent [31].*

VALORES	DESCRIPCIÓN
Values[0]	Indica la aceleración del dispositivo menos la aceleración de la gravedad en el eje *x*.
Values[1]	Indica la aceleración del dispositivo menos la aceleración de la gravedad en el eje *y*.
Values[2]	Indica la aceleración del dispositivo menos la aceleración de la gravedad en el eje *z*.

Tabla 1. Valores proporcionados por el acelerómetro

Al proporcionar la aceleración absoluta del dispositivo, se debe hacer un paso intermedio para extraer únicamente la gravedad en cada eje, indicando de esta forma su orientación en el espacio.

Para aislar la gravedad la misma documentación de Android, *SensorEvent* sugiere aplicar un filtro a paso bajo de tal manera que la gravedad sea un sumatorio ponderado entre el valor anterior (con el mayor peso), más un incremento del valor del acelerómetro con menos peso. Este método se estabiliza rápidamente entorno a los 9.81 m/s^2 de la gravedad y además no le da mucho peso a las aceleraciones puntuales y bruscas que se puedan dar en el dispositivo *[31].*

$$gravedad = \alpha \cdot gravedad + (1 - \alpha) \cdot value$$

Donde α es un valor mayor 0 y menor que 1, que cuanto más cercano esté a la unidad más restrictivo será. En las pruebas realizadas para comprobar las funciones implementadas en la clase *DistanceEstamtion* se usa el valor 0,8.

El cálculo del *Pitch* se obtiene mediante la descomposición de la gravedad, pues sus componentes se corresponden con los valores que se obtienen del acelerómetro.

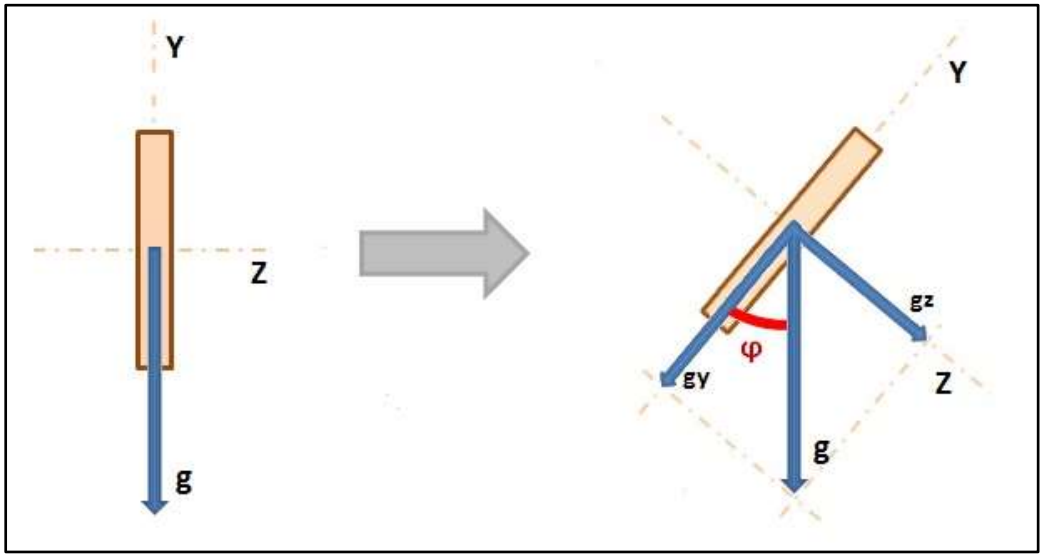


Ilustración 39. Esquema del cálculo del pitch.

$$\varphi = cos^{-1}(^{gy}/_{g}) = 90 - cos^{-1}(^{gz}/_{g})$$

4.3 Modelo PinHole

Todo el cálculo de distancias y parámetros en este proyecto está basado en el modelo *PinHole*, por lo que es necesario matizar algunos aspectos de dicho modelo.

El modelo *PinHole* simplifica el comportamiento de la óptica, reduciendo ésta a un punto (punto focal), por donde pasan todos los rayos luminosos. Para los cálculos el rayo luminoso de interés es aquel que pasa directamente por la distancia focal, y como se puede observar en la siguiente imagen, las ecuaciones se obtienen mediante triángulos semejantes que relacionan un punto del espacio con su proyección en la imagen.

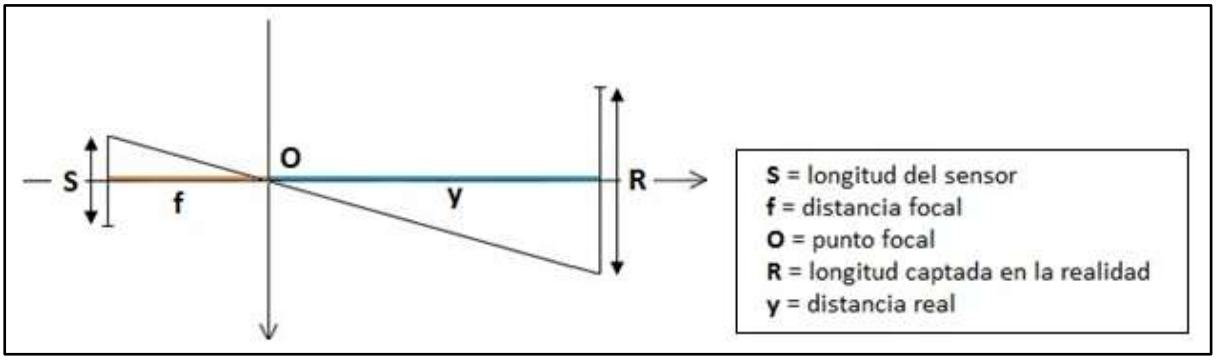


Ilustración 40. Esquema del modelo PinHole.

Como ilustra la imagen este modelo nos proporciona la relación entre los parámetros intrínsecos de la cámara ($S - f$) y el mundo real ($y - R$). Al ser dos triángulos semejantes se puede obtener la siguiente ecuación:

$$\frac{f}{S/2} = \frac{y}{R/2} \quad \rightarrow \quad \frac{f}{S} = \frac{y}{R}$$

El elemento de interés final es la distancia, por lo que para obtenerla se necesitan los términos de la igualdad situados a la izquierda, que hacen referencia a los parámetros intrínsecos y que se explicará su obtención paso a paso en el siguiente apartado. Pero también se necesita la longitud captada por la realidad. Obviamente en cualquier aplicación real este dato será desconocido puesto que el móvil se utiliza con ese objetivo, hacer un reconocimiento del entorno del cual no tenemos información previa. Es en este punto donde se introduce un nuevo parámetro que es la **altura de colocación del dispositivo**, que debe ser prefijada antes de iniciar cualquier aplicación.

Las siguientes figuras muestran cómo se introduce la altura en la anterior ecuación para sustituir el dato desconocido:

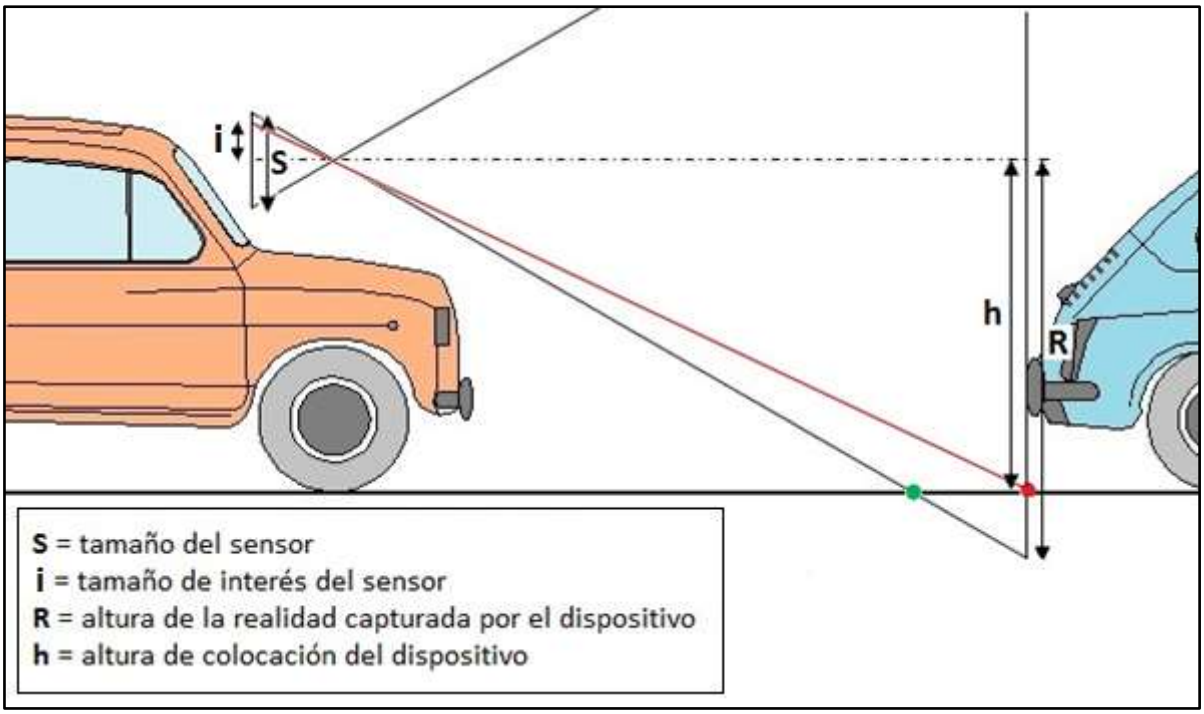


Ilustración 41. Modelo PinHole aplicado a la detección de coches I.

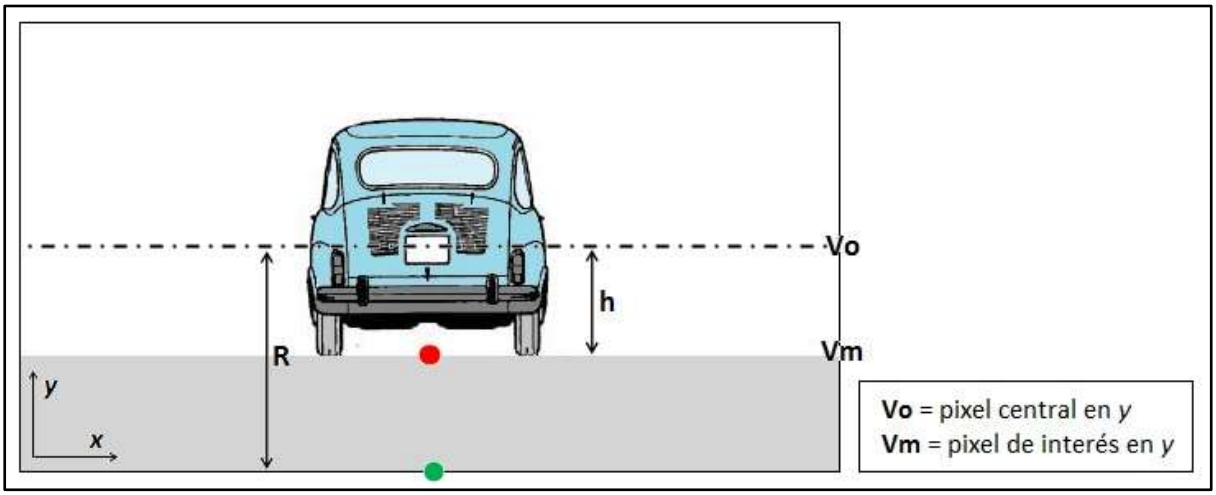


Ilustración 42. Modelo PinHole aplicado a la detección de coches II.

Ambas imágenes representan lo mismo desde distintas perspectivas: el punto rojo es el pixel de interés puesto que es donde se encuentra el siguiente vehículo (*Vm*), el punto verde representa el pixel máximo que captaría la cámara por abajo y que se corresponde

con la mitad de *R*. Por otro lado la altura *h,* vendría dada por la distancia que hay desde el suelo hasta donde se coloca el dispositivo que coincide con el eje central del sensor.

En las dos figuras se representa el caso más probable: que la altura capturada por la cámara no coincida con la altura de colocación del dispositivo, es decir:

$$h \neq R$$

El único caso donde coincidirían es cuando el suelo coincide con el punto de interés, por ejemplo, en la *ilustración 41* el vehículo tendría que estar más cerca haciendo coincidir el punto verde con el rojo. En ese único caso se cumpliría:

$$R = h \quad \rightarrow \quad \frac{f}{S/2} = \frac{y}{h}$$

Pero en el caso de que no coincidiesen se debe introducir una nueva incógnita (*i*), que hace referencia a la proporción del sensor necesaria para captar únicamente la altura. Por lo tanto la anterior ecuación quedaría de la siguiente forma:

$$\frac{f}{S/2} = \frac{y}{h} \quad \rightarrow \quad \frac{f}{i} = \frac{y}{h}$$

Este nuevo término ("i"), se obtiene mediante una regla de proporcionalidad que, aunque es sencilla matemáticamente, es algo compleja de desarrollar. Con la siguiente figura se pretende mejorar esta comprensión:

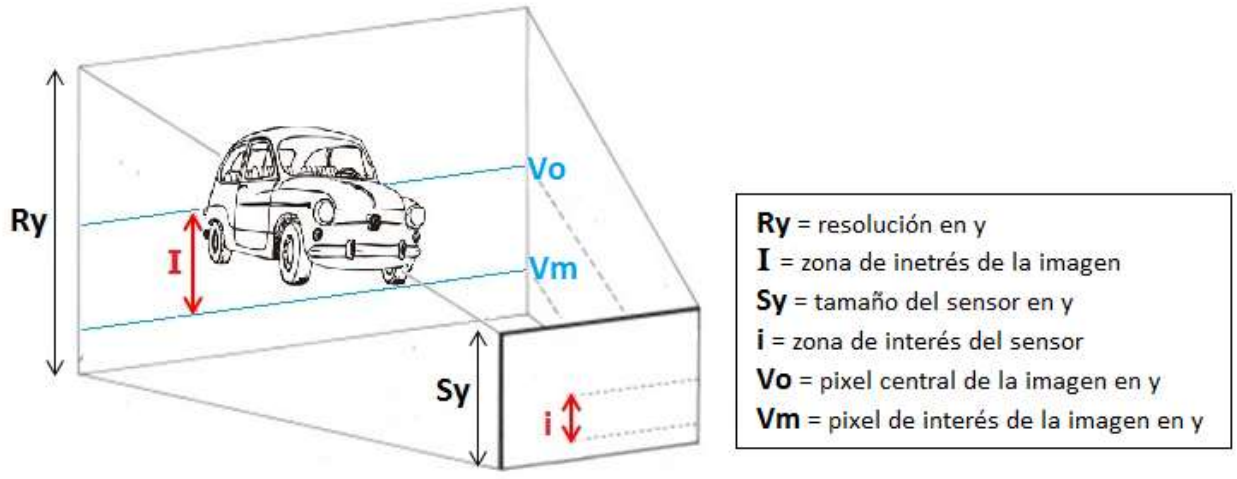


Ilustración 43. Proporción imagen-sensor.

La figura representa una comparación entre el sensor y la imagen, es decir, la altura está representada en la imagen por una determinada cantidad de pixeles, que equivalen en

el sensor a unos milímetros proporcionales. Es lógico que según la resolución de la imagen, la altura real estará representada por distinta cantidad de píxeles. Con una simple regla de tres o proporción se obtiene la relación entre píxeles y unidades métricas.

$$\frac{Ry}{I} = \frac{Sy}{i}$$

$$I = Vo - Vm$$

$$\frac{Ry}{Vo - Vm} = \frac{Sy}{i}$$

A continuación se sustituye ambas ecuaciones se despeja el término *i,* para más tarde sustituirlo en la ecuación principal:

$$\frac{Ry}{Vo - Vm} = \frac{Sy}{i} \quad \rightarrow \quad i = \frac{Sy \cdot (Vo - Vm)}{Ry}$$

$$\frac{f}{i} = \frac{y}{h} \quad \rightarrow \quad \frac{f}{\frac{Sy \cdot (Vo - Vm)}{Ry}} = \frac{y}{h} \quad \rightarrow \quad y = \frac{f \cdot h \cdot Ry}{Sy \cdot (Vo - Vm)}$$

Para los posteriores cálculos se usará esta ecuación de obtención de distancias que pasará a llamarse *yo*, ya que hace referencia a la distancia inicial, es decir la distancia sin tener en cuenta la rotación del dispositivo.

$$yo = \frac{f \cdot h \cdot Ry}{Sy \cdot (Vo - Vm)}$$

Además la forma matricial para representar al modelo pinole es la siguiente:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ h \end{bmatrix}$$

4.4 Métodos de cálculo

Por último se unifican los dos conceptos anteriores, el cálculo de distancias con el error de rotación en la imagen, para lo cual se desarrollan tres métodos que posteriormente se analizarán en casos reales.

4.4.1 Método 1: Cálculo de la distancia mediante trigonometría

Este método está obtenido a partir del libro *Visión por Computador* escrito por Arturo de la Escalera Hueso *[32],* donde se explica que conociendo el ángulo de rotación en *x*, es decir el pitch, se puede conocer la nueva distancia calculada a partir de la distancia original.

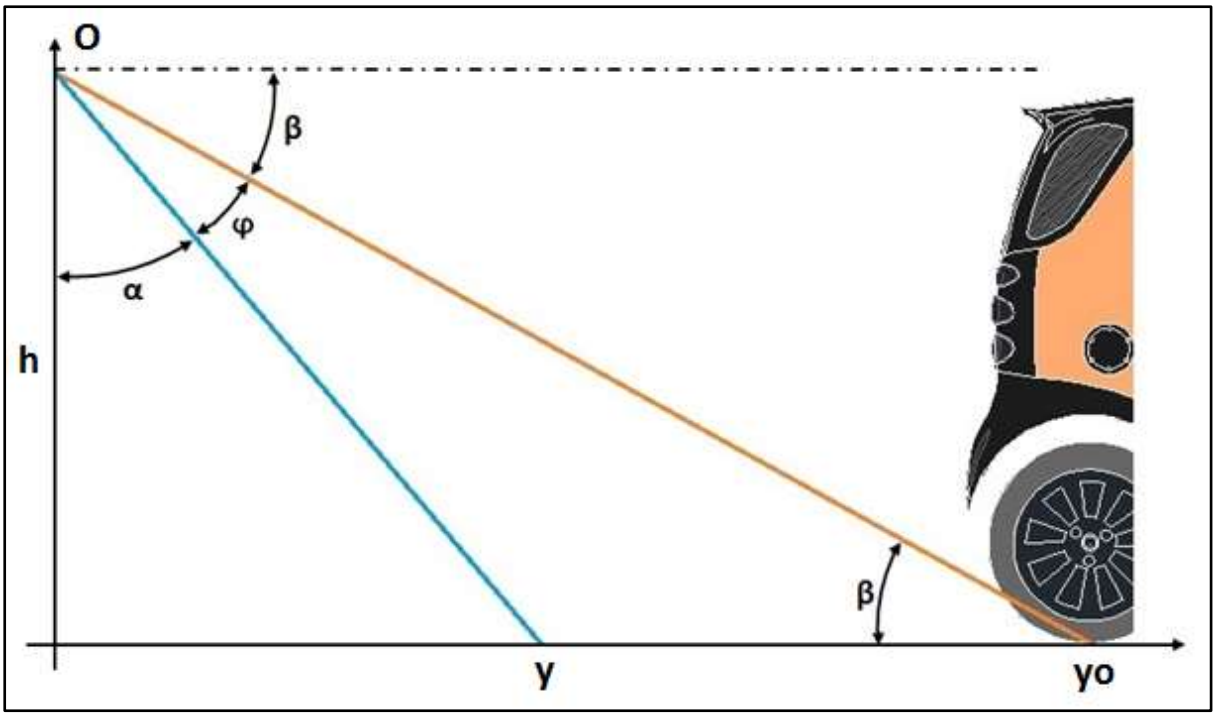


Ilustración 44. Esquema principal del cálculo mediante el método 1.

El punto *O* de la figura es donde se coloca el dispositivo por lo que la línea naranja representa el eje de captura en la posición inicial, y la línea azul representa el eje de captura de una posición tras la variación del *Pitch*. El ángulo φ, el *Pitch*, es conocido, al igual que la posición inicial, (*ecuación yo de la sección 4.3 Modelo PinHole*). También es conocida la

altura (*h*), la cual es una variable fija estimada en 1.50 metros, pues es la altura media de los vehículos, donde se prevé que debería estar colocado el dispositivo.

Conocidos la altura (*h*) y la posición inicial (*yo*) se puede obtener β:

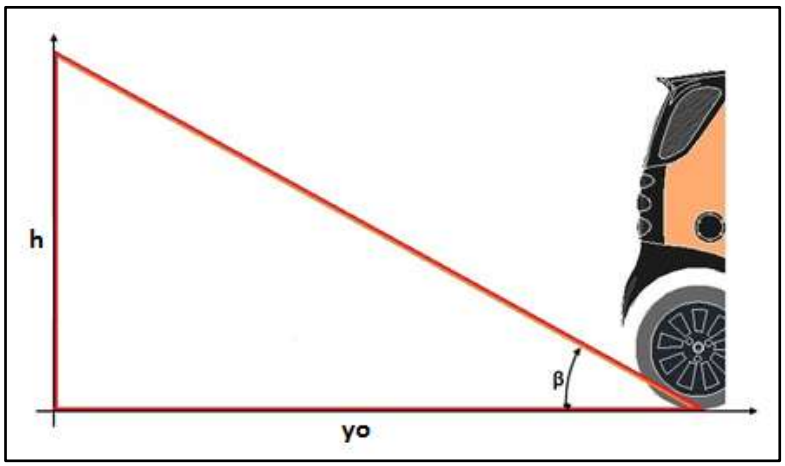


Ilustración 45. Esquema para la obtención de β.

$$\beta = \tan^{-1}\frac{h}{yo}$$

Como se puede observar en la *ilustración 44*, la suma de los tres ángulos (α, β y φ) forman un ángulo recto, y una vez conocidos β y φ obtener α es inmediato:

$$\frac{\pi}{2} = \alpha + \beta + \varphi \quad \rightarrow \quad \alpha = \frac{\pi}{2} - \beta - \varphi$$

Por último se calcula la distancia real (*y*), mediante el ángulo α y la altura (*h*):

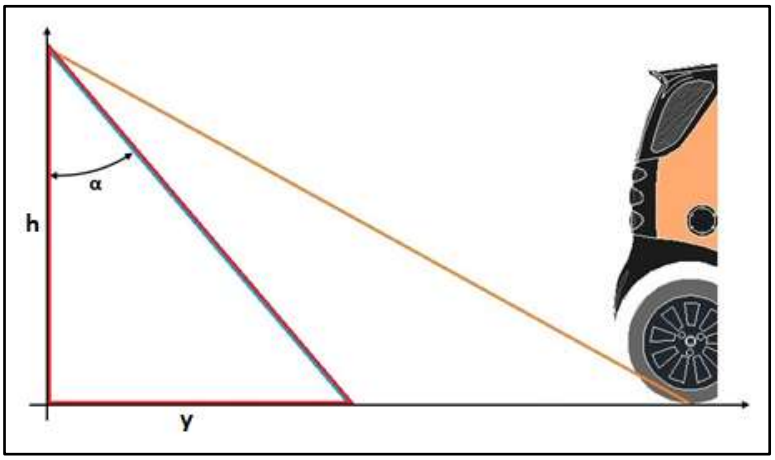


Ilustración 46. Esquema para la obtención de y.

$$y = h \cdot \tan \alpha$$

En este método se obtiene una estimación de la distancia, sabiendo los metros que separan al vehículo del dispositivo en el eje y, teniendo en cuenta la inclinación del móvil (el pitch), y con un cálculo previo de la distancia inicial mediante el modelo *PinHole*.

4.4.2 Método 2: Cálculo de la distancia mediante matrices de rotación (Ángulos de Euler)

Para este método se calcula previamente la distancia hacia el objeto de interés mediante los parámetros intrínsecos de la cámara, según la ecuación obtenida del modelo *PinHole*:

$$yo = \frac{f \cdot h \cdot Ry}{Sy \cdot (Vo - Vm)}$$

A partir de este dato se aplican las tres matrices de rotación (*Pitch, Roll y Yaw*), que modificarán las coordenadas en el espacio para proporcionar otro eje en el que ya están incluidas dichas variaciones, es decir, como se sabe la rotación que se ha producido en el móvil y estas no son deseables por las deformaciones anteriormente vistas, se convierte el sistema rotado en el original, aplicando dichas matrices de rotación:

$$Yaw = \theta = \begin{bmatrix} \cos(\Delta\theta) & 0 & sen(\Delta\theta) \\ 0 & 1 & 0 \\ -sen(\Delta\theta) & 0 & \cos(\Delta\theta) \end{bmatrix}$$

$$Pitch = \varphi = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \cos(\Delta\varphi) & -sen(\Delta\varphi) \\ 0 & sen(\Delta\varphi) & \cos(\Delta\varphi) \end{bmatrix}$$

$$Roll = \delta = \begin{bmatrix} \cos(\Delta\delta) & -sen(\Delta\delta) & 0 \\ sen(\Delta\delta) & \cos(\Delta\delta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Aunque más tarde se verá que las matrices de rotación también están incluidas en la clase *DistanceEstimation*, el elemento de interés principal en estas ecuaciones es la distancia ("y"). Para su cálculo, aunque en las ecuaciones se observa que depende de los tres ángulos de Euler, únicamente se tendrá en cuenta el pitch, por las razones ya explicadas en la *sección 4.2.1 Acelerómetro: caculo del Pitch (pág 24)*. A continuación se desarrolla la obtención de la ecuación de la distancia para analizarla por separado:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \cos(\Delta\varphi) & -sen(\Delta\varphi) \\ 0 & sen(\Delta\varphi) & \cos(\Delta\varphi) \end{bmatrix} \begin{bmatrix} xo \\ yo \\ h \end{bmatrix}$$

$$y = yo \cdot \cos(\Delta\varphi) - h \cdot sen(\Delta\varphi)$$

Dicha distancia dependerá de la distancia inicial, calculada sin tener en cuenta las rotaciones, y de la altura a la que se coloca el dispositivo. Ambos términos multiplicados por un factor de corrección para obtener la distancia real hacia el objeto sin que las deformaciones sufridas en la imagen afecten el resultado real.

4.4.3 Método 3: Cálculo de la distancia mediante proporciones

Este método se basa en el modelo *PinHole*; cuando aplicamos una inclinación al dispositivo el eje de este modelo también sufriría dicha inclinación, como se muestra en la siguiente imagen.

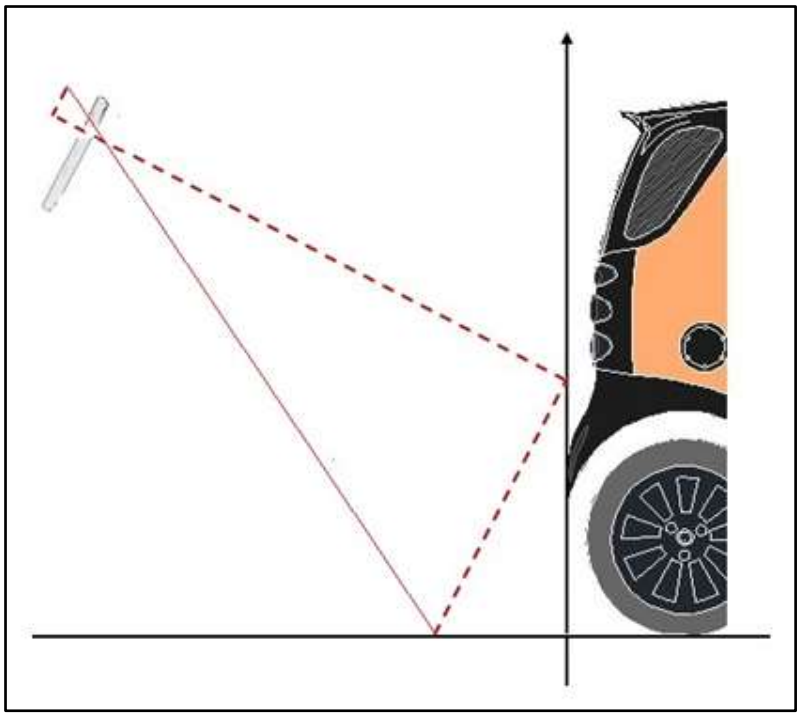


Ilustración 47. Esquema del modelo PinHole con inclinación.

La escala entre la parte que hace referencia al sensor y la parte que hace referencia a la realidad es tan grande, que para poder observar ambas juntas no se han respetado las proporciones. Se observa que la ecuación del modelo *PinHole* se complica, puesto que la distancia real no coincide con el eje del modelo *Pinhole*, al igual que la altura.

Para la explicación de este método, en las siguientes figura solo se representará la parte del modelo que hace referencia a la realidad, de esta manera será más sencillo explicar la relación encontrada entre el modelo *Pinhole* sin y con inclinación.

Conociendo el ángulo de inclinación, el *Pitch* (φ), la altura a la que se coloca el dispositivo (*h*), y utilizando la ecuación del modelo *PinHole* con la relación de pixeles-longitud incluida, se puede obtener la distancia real hacia el objeto de interés.

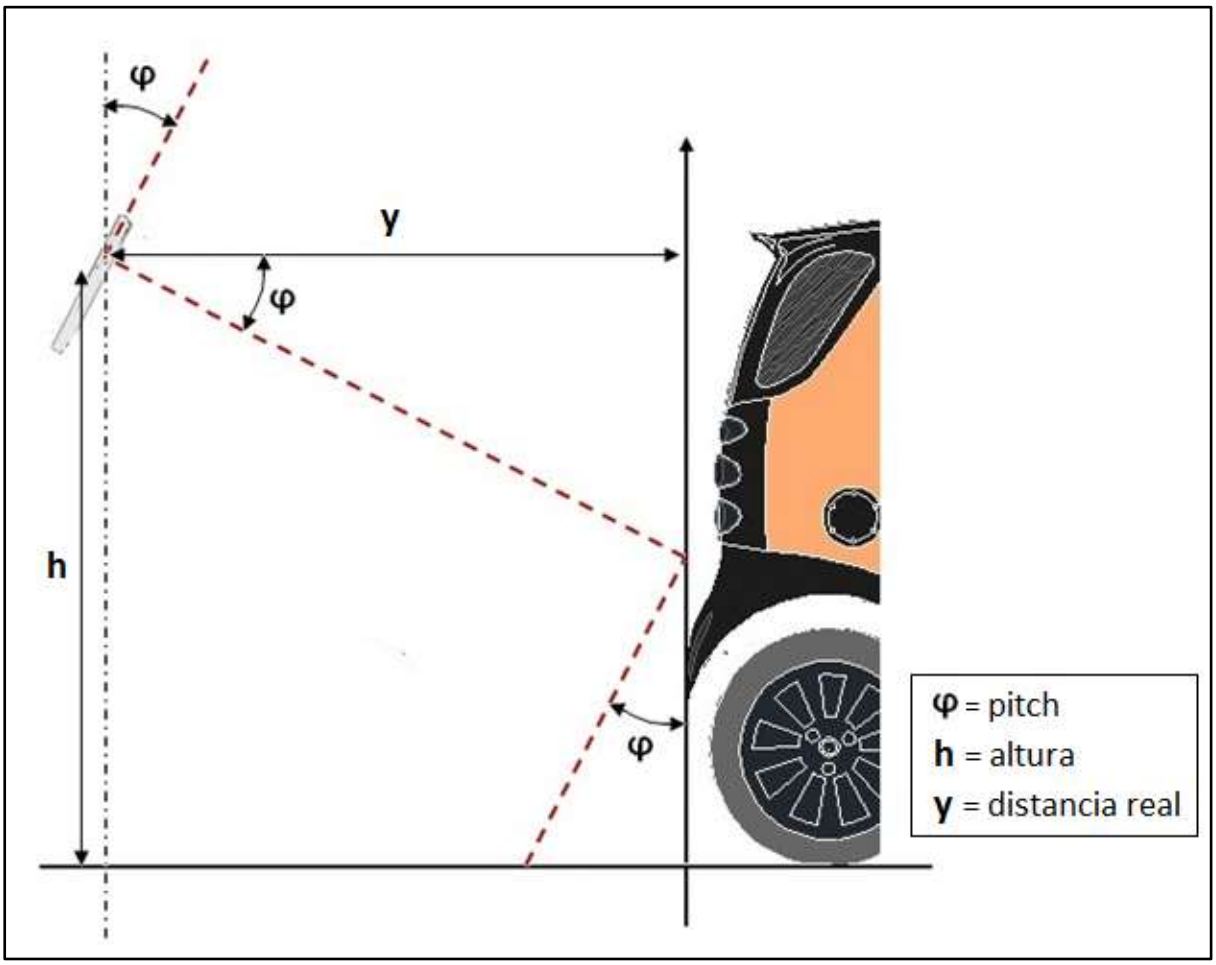


Ilustración 48. Esquema del modelo PinHole con inclinación, ampliado.

En la figura se observan dos triángulos rectángulos proporcionales:

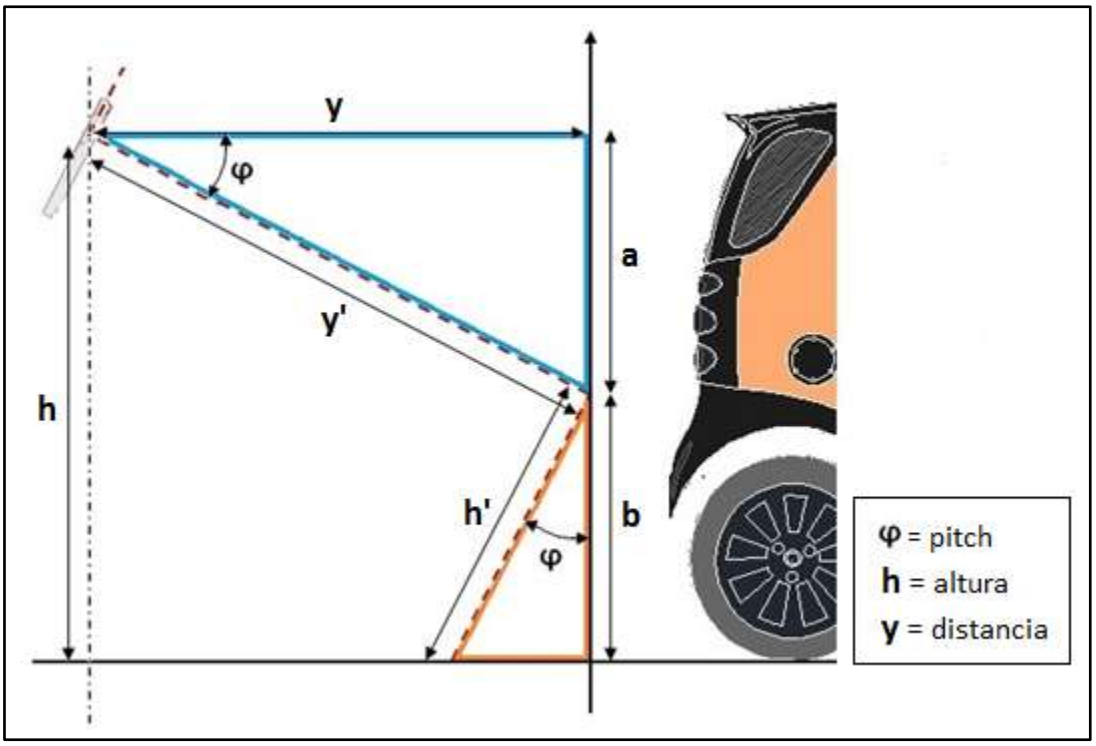


Ilustración 49. Esquema de los triángulos semejantes.

Partiendo de la ecuación del modelo *PinHole*, se hace una semejanza con los nuevos datos:

$$\frac{f}{i} = \frac{y}{h} \quad \rightarrow \quad \frac{f}{i} = \frac{y'}{h'}$$

Del triángulo azul se obtiene:

$$\cos\varphi = \frac{y}{y'} \quad \rightarrow \quad y' = \frac{y}{cos\varphi}$$

Del triángulo naranja se obtiene:

$$cos\varphi = \frac{b}{h'} \quad \rightarrow \quad h' = \frac{b}{cos\varphi}$$

Sustituyendo ambas ecuaciones en la primera se obtiene:

$$\frac{f}{i} = \frac{\frac{y}{cos\varphi}}{\frac{b}{cos\varphi}} = \frac{y}{b}$$

Para obtener *b*, se vuelve a retomar la *ilustración 49*, de donde se deduce la siguiente igualdad referente a la altura:

$$h = a + b \quad \rightarrow \quad b = h - a$$

Para obtener *a*, se vuelve a usar el triángulo azul, de donde se obtiene:

$$tan\varphi = \frac{a}{y} \quad \rightarrow \quad a = y \cdot tan\varphi$$

Y sustituyendo *a* en la ecuación de *b*, se obtiene:

$$b = h - y \cdot tan\varphi$$

Por último se sustituye esta igualdad en la ecuación principal y se despeja la *y*, pues es el término buscado:

$$\frac{f}{i} = \frac{y}{h - y \cdot tan\varphi} \quad \rightarrow \quad y = \frac{f \cdot (h - y \cdot tan\varphi)}{i}$$

$$i = \frac{Sy \cdot (Vo - Vm)}{Ry}$$

$$y = \frac{Ry \cdot \frac{f}{Sy} \cdot h}{(Vo - Vm) + Ry \cdot \frac{f}{Sy} \cdot tan\varphi}$$

4.4.4 Cálculo de la distancia horizontal

Para poder calcular la componente *x* se necesita saber la distancia a la que se encuentra el vehículo (*y*), por lo que será el último paso a realizar.

La *x* se puede calcular despejando la componente en la matriz del modelo *PinHole*.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ h \end{bmatrix}$$

$$x = \frac{y(Um - Uo)}{f_x}$$

Siendo *Um* y *Uo* los píxeles en el plano horizontal de la imagen. *Um* es el pixel de interés en el eje *x, Uo* el pixel central en el eje *x* y la distancia *y* debe ser calculada previamente por alguno de los anteriores métodos.

4.5 Implementación: descripción de la clase

La importancia de crear una clase reside en que permite agrupar elementos relacionados, así como controlar la visibilidad y accesibilidad en los posteriores procesos donde serán usados. Esta encapsulación facilita la comprensión y el mantenimiento de las futuras aplicaciones.

La clase se denomina *DistanceEstimation* y está formada por los siguientes atributos y métodos:

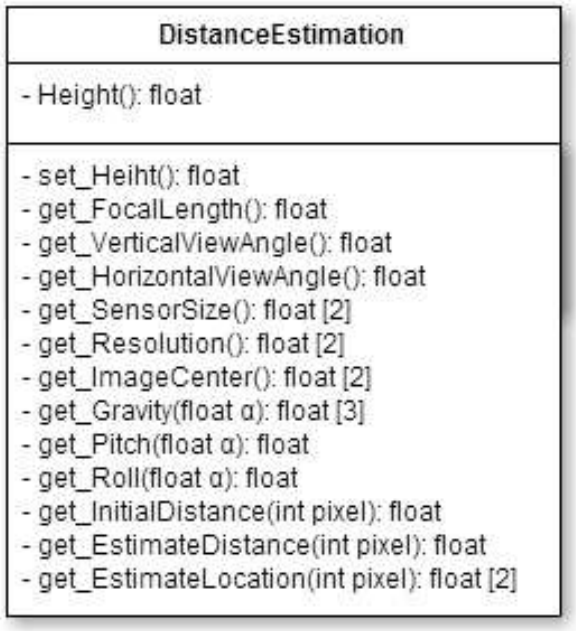


Ilustración 50. Diagrama UML de DistanceEstimation.

4.5.1 Atributos

Height_ float

Se trata de una constante que indica la altura a la que se encuentra colocado el dispositivo. Será necesaria para llevar a cabo muchos de los cálculos usados en las funciones y tendrá que ser ajustada manualmente, ya que es imposible determinar a qué altura se encuentra el dispositivo usando sus sensores.

4.5.2 Métodos

set_Heigth_(): float

Esta función permite ajustar la altura de colocación del dispositivo, en caso de que sea necesario modificarla. Por defecto está fijada 1,50m.

get_FocalLength(): float

Esta función devolverá la distancia focal de la cámara incorporada en el dispositivo, en milímetros.

get_VerticalViewAngle(): float

Esta función devuelve el ángulo de apertura de visión de la cámara del dispositivo en el plano vertical, en grados.

get_HorizontalViewAngle(): float

Esta función devuelve el ángulo de apertura de visión de la cámara del dispositivo en el plano horizontal, en grados.

get_SensorSize(): float [2]

Devuelve un vector que proporciona el tamaño del sensor de la cámara del dispositivo en sus componentes horizontal y vertical, en milímetros.

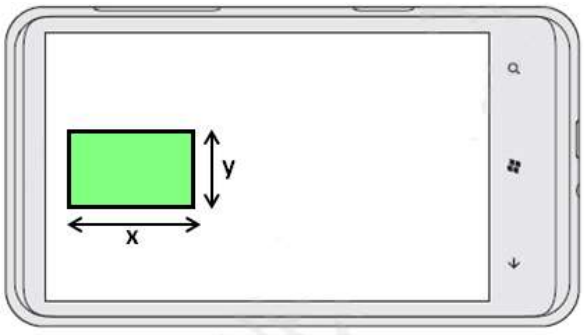


Ilustración 51. get_SensorSize().

- *Value[0]:* Devuelve el ancho del sensor estando el dispositivo colocado de manera horizontal, es decir, se corresponde con la *x* de la *ilustración 51.*
- *Value[1]:* Devuelve el alto del sensor estando el dispositivo colocado de manera horizontal, es decir, se corresponde con la *y* de la *ilustración 51.*

get_Resolution(): float [2]

Devuelve un vector que proporciona la resolución, en las dos componentes, con la que está trabajando la cámara en ese instante, en píxeles.

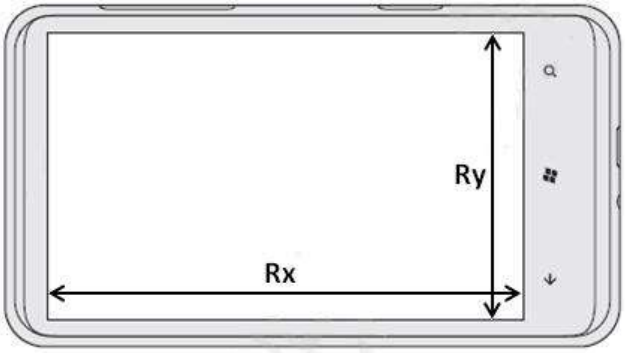


Ilustración 52. get_Resolution().

- *Value[0]:* Devuelve la resolución *Rx* de la *ilustración 52.*
- *Value[1]:* Devuelve la resolución *Ry* de la *ilustración 52.*

get_ImageCenter(): float [2]

Devuelve un vector que muestra las dos componentes del pixel central de la imagen de acuerdo con la resolución en la cual se está trabajando.

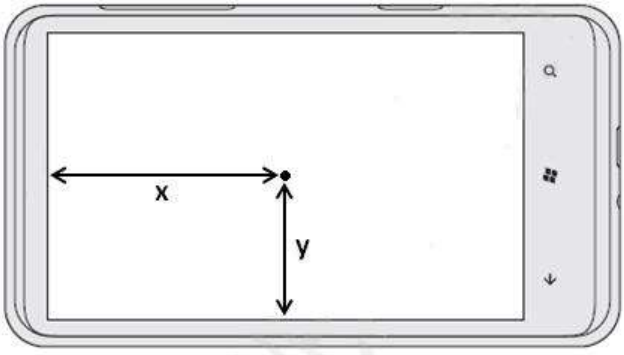


Ilustración 53. get_ImageCenter().

- *Value[0]:* Devuelve la resolución correspondiente a la *x* en la *ilustración 53,* es decir, la mitad de *Rx.*
- *Value[1]:* Devuelve la resolución correspondiente a la *y* en la *ilustración 53,* es decir, la mitad de *Ry.*

get_Gravity(α): float [3]

Devuelve un vector que indica la gravedad, a la que está sometido el dispositivo, descompuesta en sus tres ejes de coordenadas, en metros por segundo al cuadrado.

El parámetro α indica el grado de restricción del filtro a paso bajo que usa la función para poder aislar la gravedad. El rango oscila de 0 a 1, siendo más restrictivo según se acerque a 1.

get_Pitch(α): float

Esta función es una ampliación de la función *get_Gravity(α)* en la que se usa la gravedad descompuesta para calcular el Pitch al que se encuentra el dispositivo, en grados.

El parámetro α es el mismo que para la función anterior.

get_Roll(α): float

Esta función es una ampliación de la función *get_Gravity(α)* en la que se usa la gravedad descompuesta para calcular el Roll al que se encuentra el dispositivo, en grados.

El parámetro α es el mismo que para la función anterior.

get_InitialDistance(pixel): float

Devuelve la distancia, en metros, a la cual se encuentra el objeto que estamos buscando en la imagen de nuestro dispositivo. Esta función no tiene en cuenta los posibles errores cometidos debido a las rotaciones. Se necesita el parámetro pixel, ya que se trata del pixel en el cual se encuentra situado el objeto de interés, variable en cada caso.

get_EstimateDistance(pixel); float

Devuelve la distancia, en metros, a la cual se encuentra el objeto que estamos buscando en la imagen de nuestro dispositivo. Esta función tiene en cuenta las rotaciones a las que está sometido el dispositivo, se debe elegir entre los tres métodos de cálculo. El parámetro pixel se lo tendremos que proporcionar a la función ya que se trata del pixel en el cual se encuentra situado el objeto de interés, variable en cada caso.

get_EstimateLocation(pixel): float [2]

Devuelve un vector que muestra la distancia de las dos coordenadas, en metros, a la cual se encuentra el objeto que estamos buscando en la imagen de nuestro dispositivo. Esta función tiene en cuenta las rotaciones a las que está sometido el dispositivo, y también está por triplicado para cada uno de los cálculos.

La componente *x* tendrá su origen en el centro de la imagen, lo que permite una localización del objeto total en el plano del suelo.

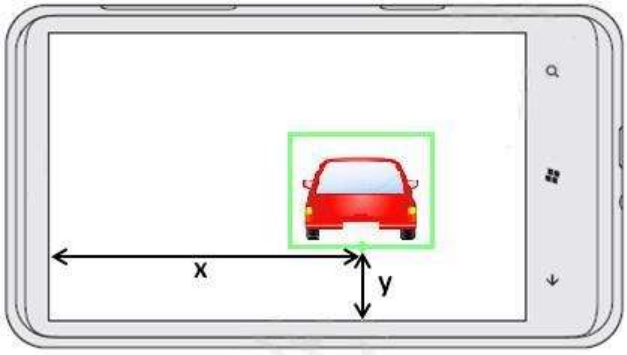


Ilustración 54. get_EstimateLocation(pixel).

- *Value[0]:* Devuelve la distancia *x*, representada en la *ilustración 54.*
- *Value[1]:* Devuelve la distancia *y,* representada en la *ilustración 54.*

Esta función tiene en cuenta las rotaciones a las que está sometido el dispositivo, y también está por triplicado para cada uno de los cálculos.

Se necesita el parámetro pixel, ya que se trata del pixel en el cual se encuentra situado el objeto de interés, variable en cada caso.

5. PRUEBAS DE EVALUACIÓN Y RESULTADOS

Aunque la clase que se ha implementado tiene numerosas funciones de obtención de parámetros, la función clave es la de obtención de distancias ya que es la más compleja y recoge la mayoría de las otras funciones. Por este motivo las pruebas para evidenciar y justificar que los cálculos son correctos y la precisión que tienen, se ha hecho, con cada uno de los tres métodos, la obtención de la distancia tanto en x como en y:

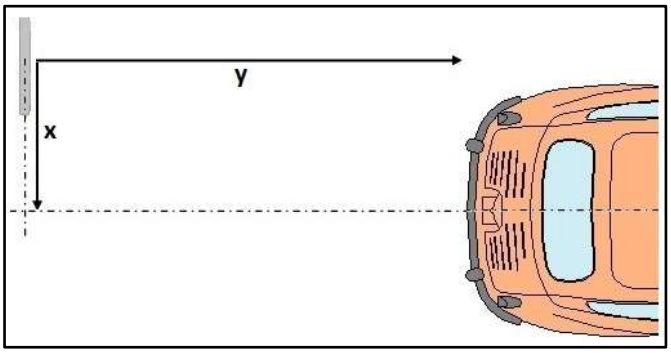


Ilustración 55. Obtención de x e y.

Para llevar a cabo una buena evaluación los test se han realizado con tres Smartphones de diferentes gamas:

- Un Sony Xperia E, representando a los dispositivos de gama baja.
- Un Jiayu G4-A, de gama media.
- Un Samsung Galaxy S3, de gama alta.

La altura a la que se coloca el dispositivo está programada a 1,50 metros, por lo que se ha usado un dispositivo fijado a dicha altura, para evitar los errores de cálculo en las imágenes capturadas.

Ilustración 56. Soporte para fijar la altura del smartphone.

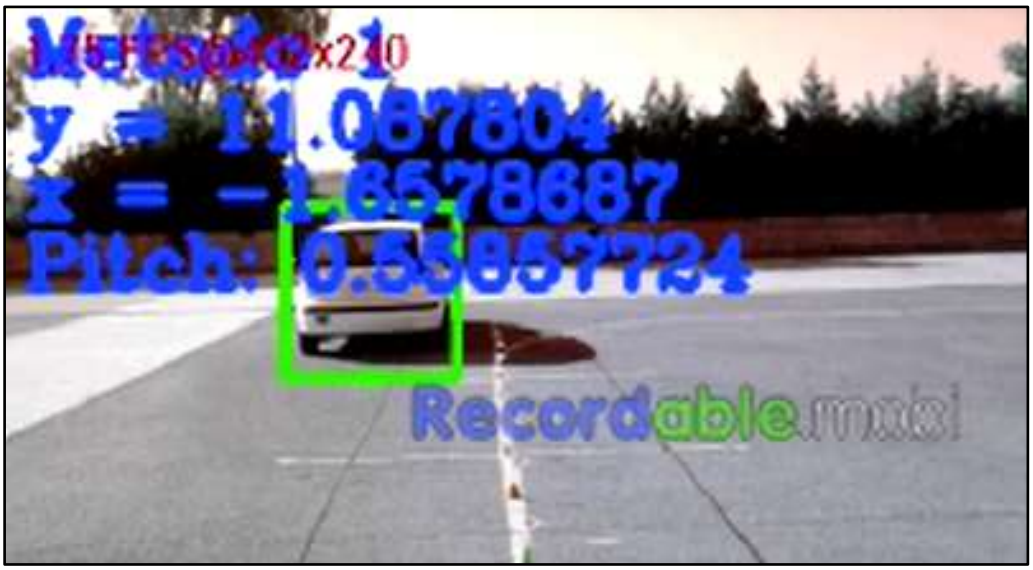
El programa usado para la detección de coches no está perfeccionado y detecta de manera muy intermitente, por este motivo, las pruebas a realizar no superan la distancia de diez metros. El método de recogida de datos consiste en acotar un recorrido de 12 metros, señalizando con líneas blanco los metros 3, 6, 9 y 12 y una línea negra colocada a un metro por cada lado del eje central. De esta manera, una vez capturada la imagen se puede comparar entre la distancia real, señalizada con las líneas y las banderillas, y la distancia estimada proporcionada por el dispositivo.

Para llevar a cabo una comparación exhaustiva se ha intentado capturar tres distancias con tres inclinaciones distintas en cada uno de los dispositivos, pero debido al ángulo de visión de cada cámara, alguna de las tomas no se han podido obtener, principalmente aquellas con las distancias y ángulos de visión más pronunciados.

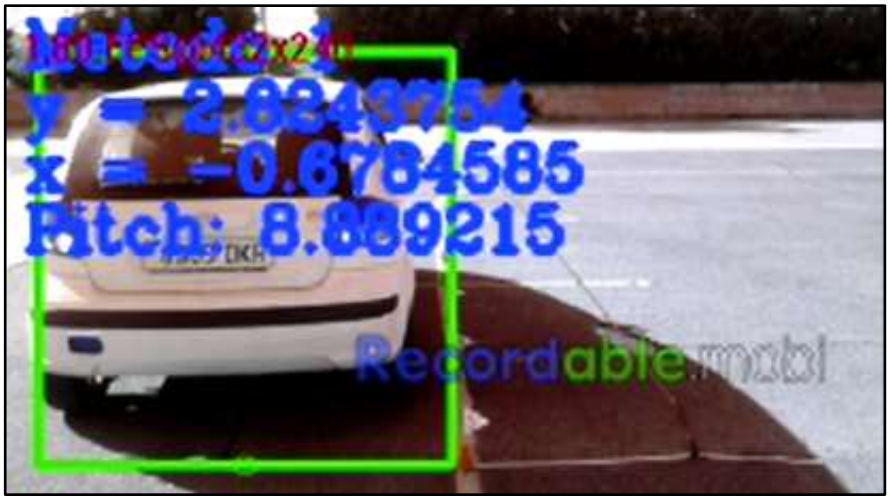
5.1 Sony Xperia E, dispositivo de gama baja

Método 1

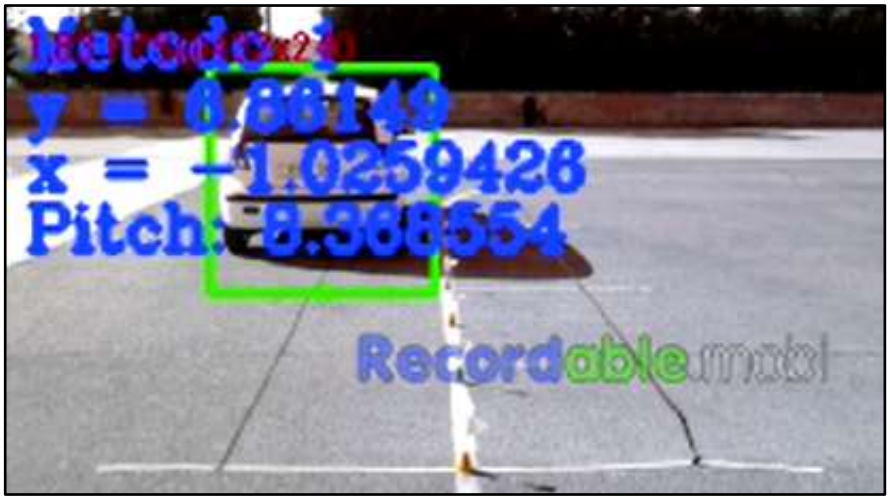
Distancia real de 6 metros.



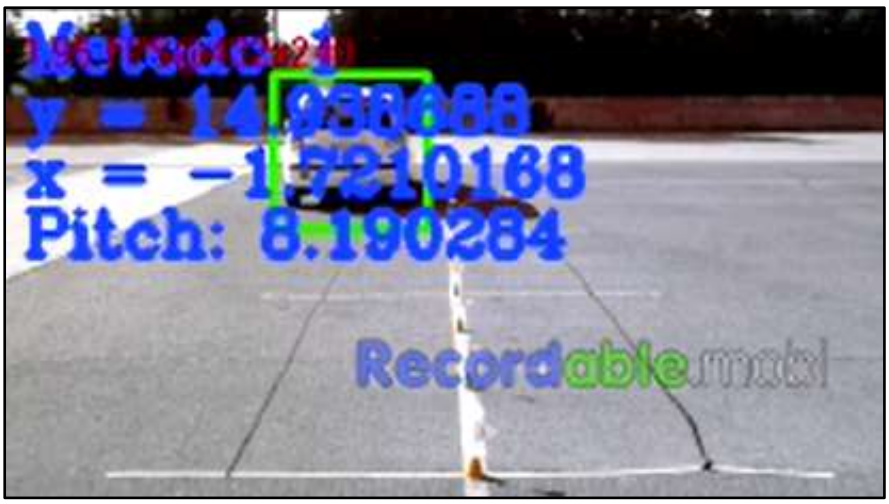
Distancia real de 9 metros.



Distancia real de 3 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Distancia real de 2,90 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	-	-	-	-	-	7,83%
	0,74	6	5,97	0,03	0,5%	
	0,56	9	11,07	2,07	23%	
PITCH MEDIO	8,89	3	2,82	0,18	6%	28,78%
	8,37	6	6,86	0,86	14,33%	
	8,16	9	14,94	5,94	66%	
PITCH ALTO	14,65	2,90	2,56	0,34	11,72%	11,72%
	-	-	-	-	-	
	-	-	-	-	-	

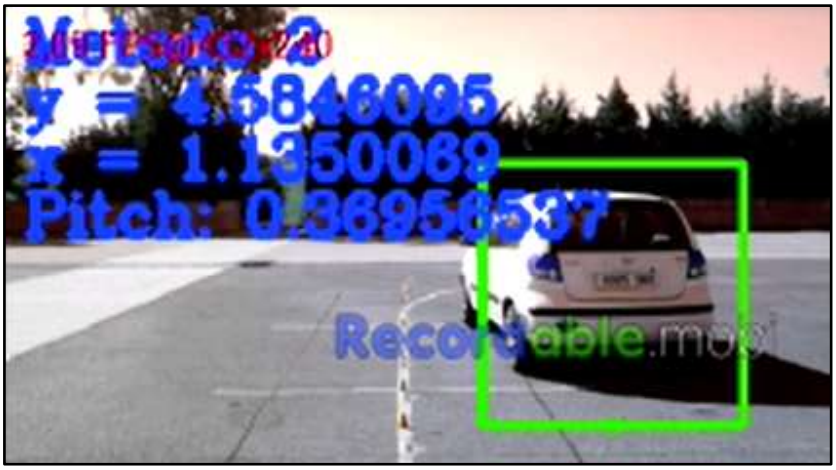
Tabla 2. Dispositivo de gama baja, método 1.

	Error medio
DISTANCIA CORTA	8,86%
DISTANCIA MEDIA	7,41%
DISTANCIA LARGA	44,50%

Tabla 3. Dispositivo de gama baja, método 1, general.

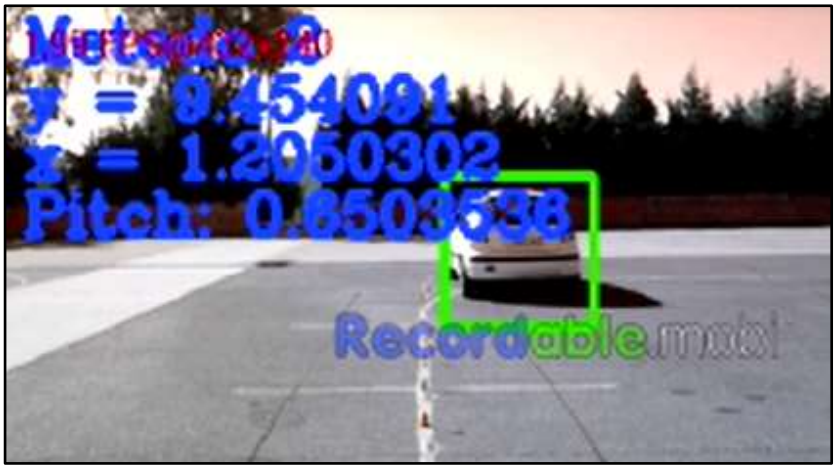
El método 1, aplicado a un dispositivo de gama baja tiene un error global en torno al 16%. Un error aceptable pero que empeorando a medida que el punto de interés se aleja, debido a la perdida de precisión por la densidad de píxeles. Es decir, cuanto más lejos esté la zona de interés menos píxeles representarán esta zona en la imagen, con la consiguiente pérdida de precisión. En un móvil de baja gama este problema se acentuará por la resolución de la cámara.

Método 2



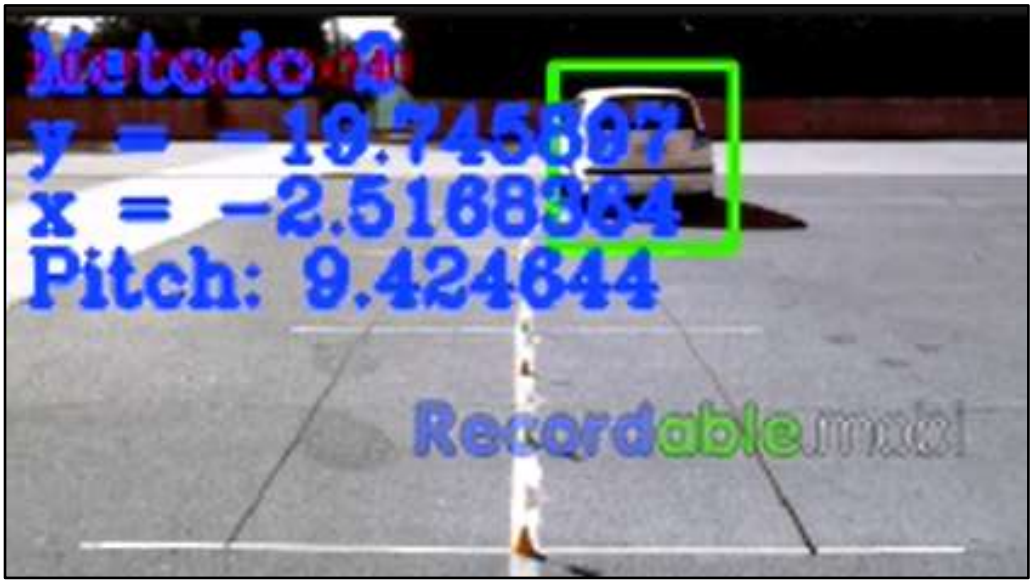
Distancia real de 5 metros.

Distancia real de 6 metros.

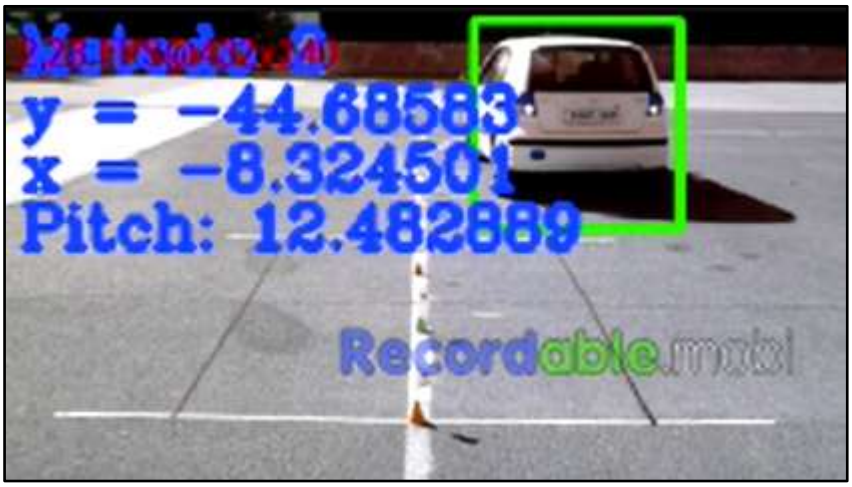


Distancia real de 9 metros.

Distancia real de 6 metros.



Distancia real de 9 metros.



Distancia real de 6,20 metros.

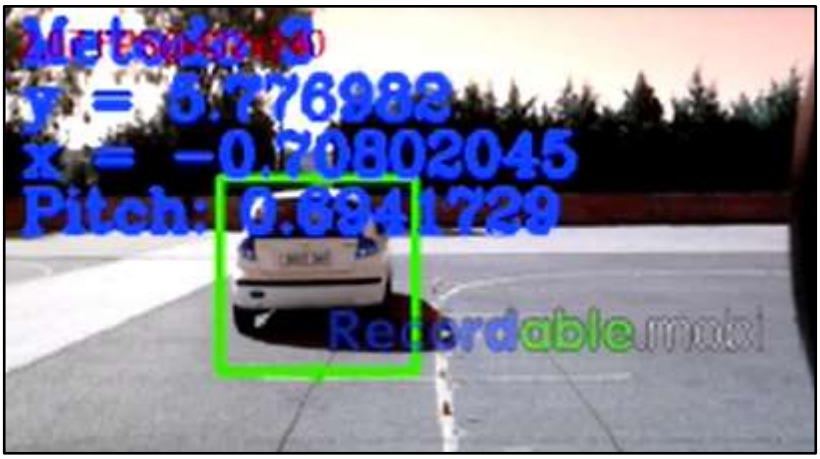
Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	*0,37*	*5*	*4,58*	*0,42*	*8,4%*	*4,79%*
	0,10	*6,20*	*6,14*	*0,06*	*0,97%*	
	0,65	*9*	*9,45*	*0,45*	*5%*	
PITCH MEDIO	*9,86*	*4,30*	*10,16*	*5,86*	*136,28%*	*295,67%*
	8,76	*6*	*31,87*	*25,87*	*431,17%*	
	9,42	*9*	*-19,76*	*28,76*	*319,56%*	
PITCH ALTO	*-*	*-*	*-*	*-*	*-*	*820,81%*
	12,48	*6,20*	*-44,69*	*50,89*	*820,81%*	
	-	*-*	*-*	*-*	*-*	

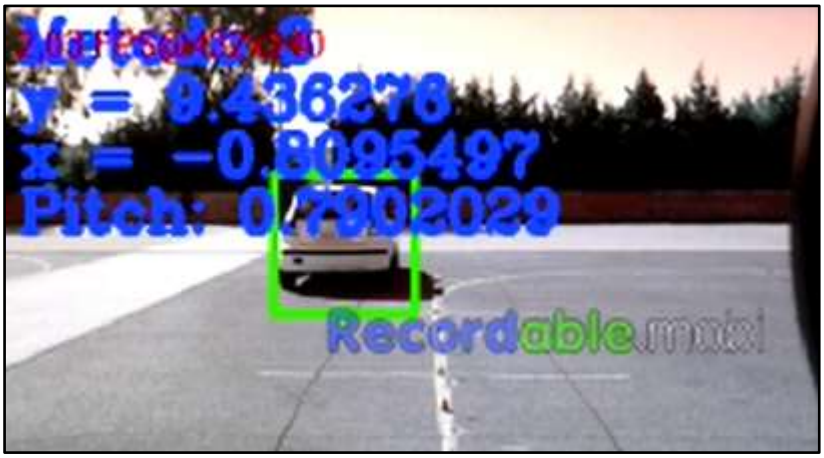
Tabla 4. Dispositivo de gama baja, método 2.

Para este caso no se calcula el error medio dependiendo de las distancias porque con esta tabla es suficiente para afirmar que este método no es eficaz cuando se le aplica pitch, por el contrario da unos errores muy pequeños cuando no se inclina el dispositivo.

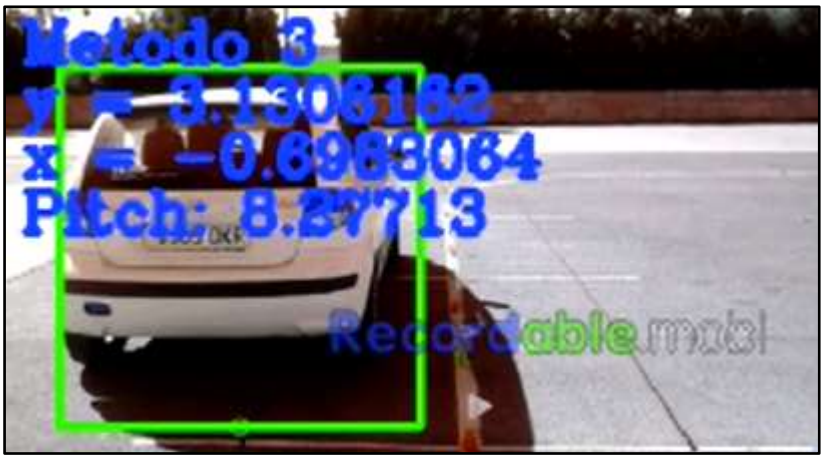
Método 3



Distancia real de 6 metros.



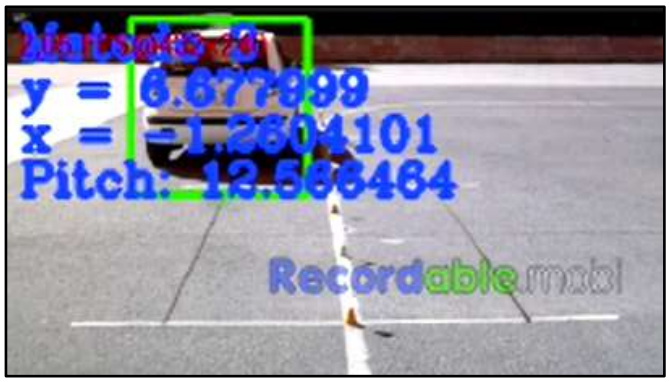
Distancia real de 9 metros.



Distancia real de 3,10 metros.

Distancia real de 5,70 metros.

Distancia real de 9 metros.



Distancia real de 5,80 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	-	-	-	-	-	*4,28%*
	0,69	*6*	*5,78*	*0,22*	*3,67%*	
	0,79	*9*	*9,44*	*0,44*	*4,89%*	
PITCH MEDIO	*8,28*	*3,10*	*3,13*	*0,03*	*0,98%*	*14,14%*
	8,21	*5,70*	*6,08*	*0,38*	*6,67%*	
	8,17	*9*	*12,13*	*3,13*	*34,78%*	
PITCH ALTO	-	-	-	-	-	*15,17%*
	12,57	*5,80*	*6,68*	*0,88*	*15,17%*	
	-	-	-	-	-	

Tabla 5. Dispositivo de gama baja, método 3.

	Error medio
DISTANCIA CORTA	*0,98%*
DISTANCIA MEDIA	*8,5%*
DISTANCIA LARGA	*19,83%*

Tabla 6. Dispositivo de gama baja, método 3, general.

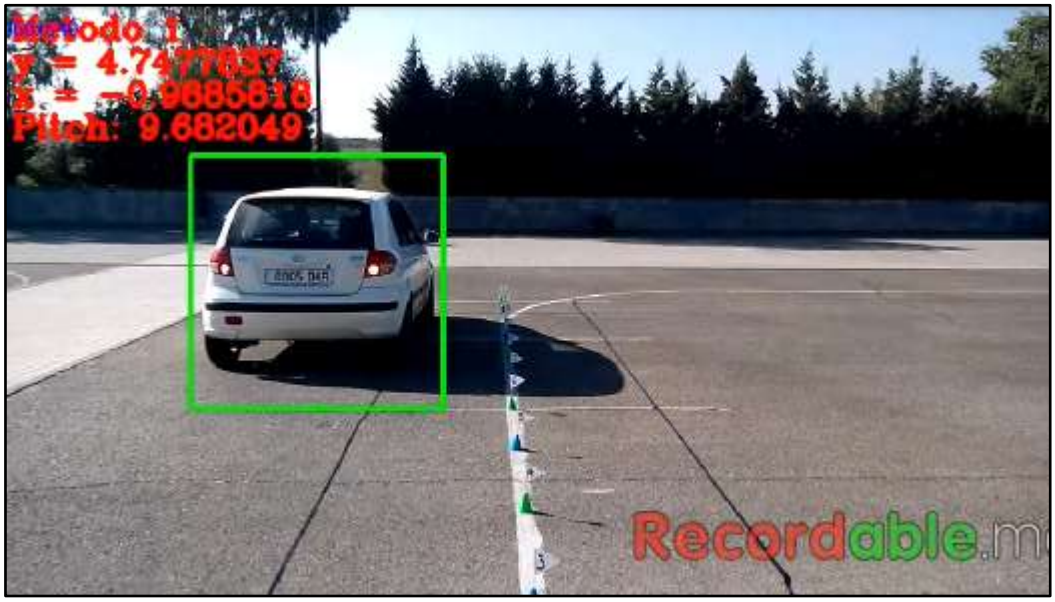
El error global del método 3 es de aproximadamente un 11%. En este caso se aprecia mejor que al aumentar la distancia del punto de interés el error también aumenta, como es lógico, llegando a tener errores muy pequeños, en torno al 1% en distancias cortas.

5.2 Jiayu G4-A, dispositivo de gama media

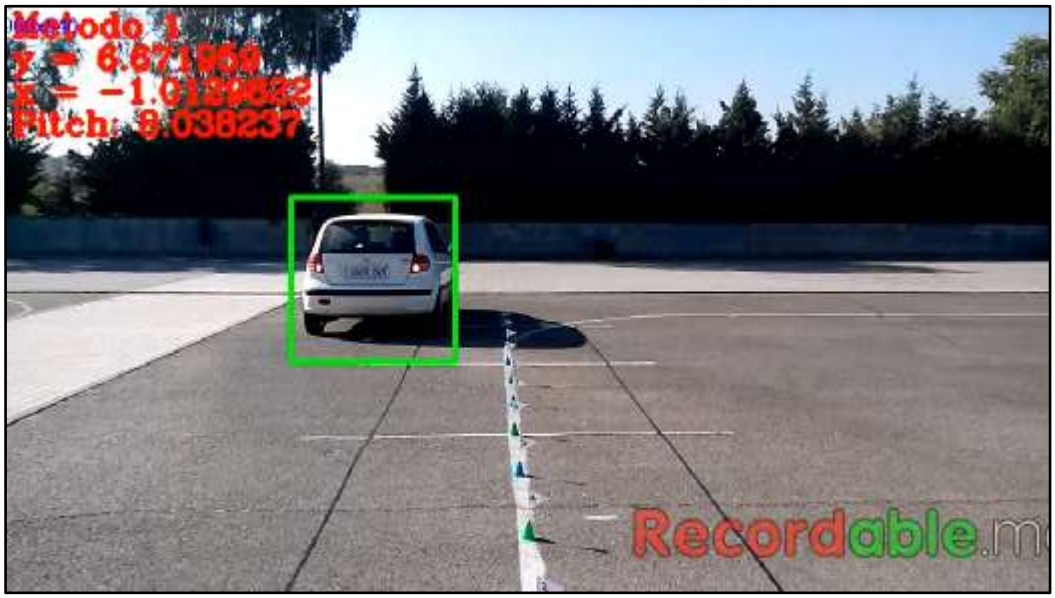
Método 1

Distancia real de 6 metros.

Distancia real de 9 metros.

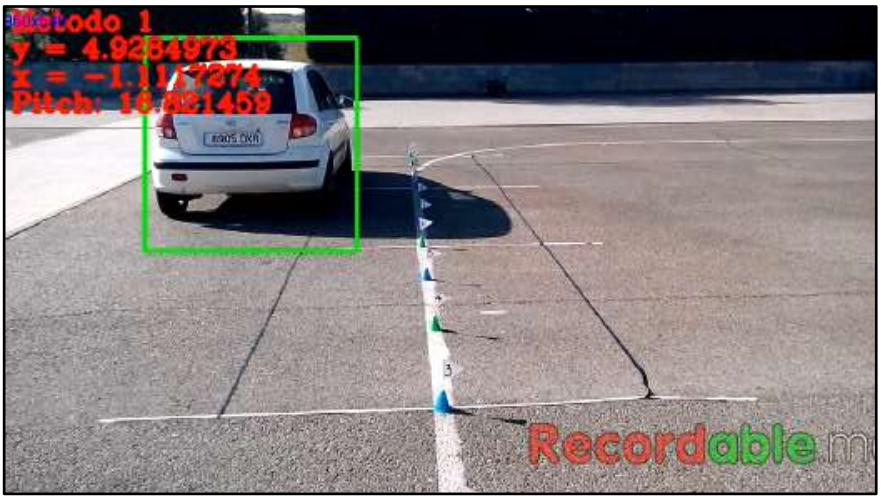


Distancia real de 6 metros.

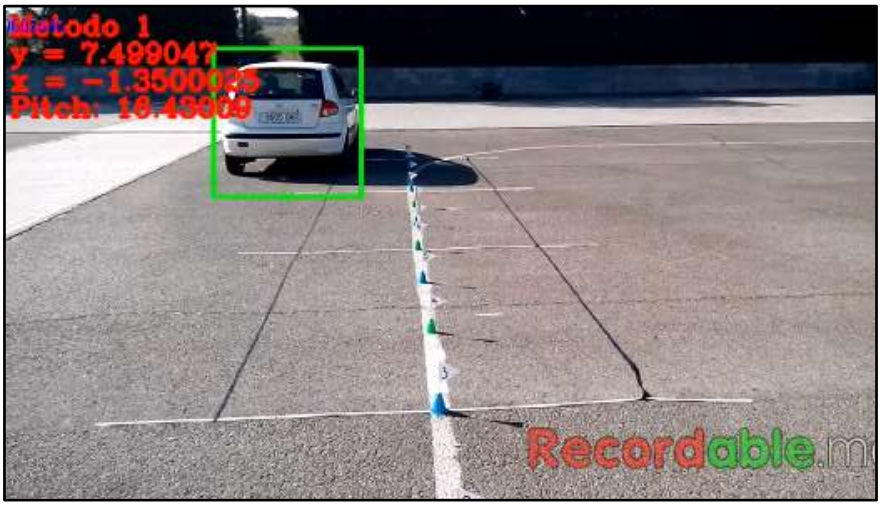


Distancia real de 9 metros.

Distancia real de 3,70 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	-	-	-	-	-	36%
	0,63	6	4,30	2,30	38,33%	
	0,23	9	6,03	3,03	33,67%	
PITCH MEDIO	-	-	-	-	-	23,30%
	9,68	6	4,75	1,25	20,83%	
	8,04	9	6,67	2,33	25,89%	
PITCH ALTO	17,59	3,70	3,19	0,51	13,78%	16,09%
	16,82	6	4,93	1,07	17,83%	
	16,43	9	7,50	1,50	16,67%	

Tabla 7. Dispositivo de gama media, método 1.

	Error medio
DISTANCIA CORTA	13,78%
DISTANCIA MEDIA	25,66%
DISTANCIA LARGA	25,41%

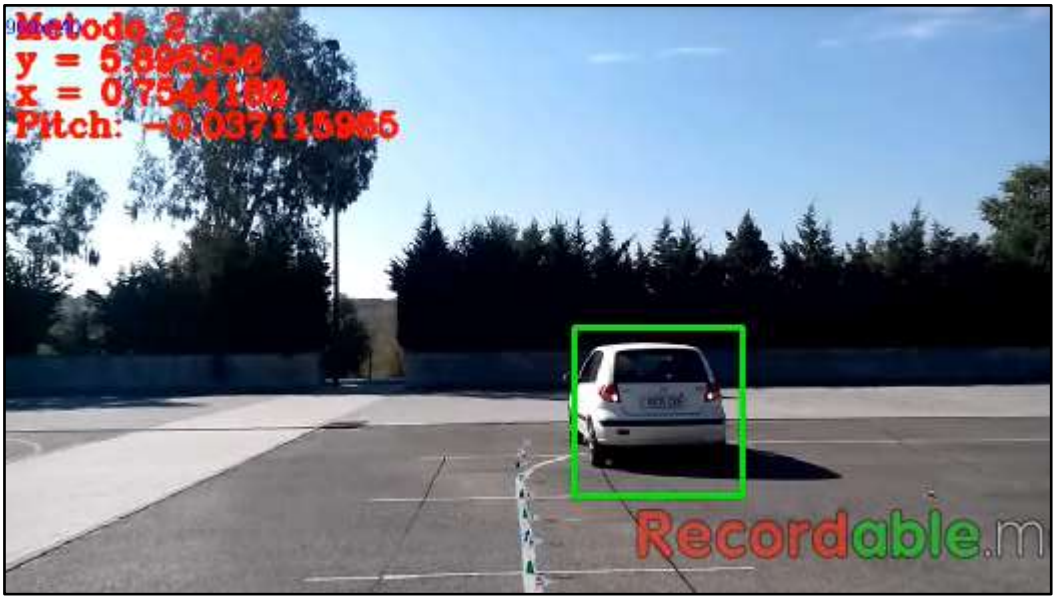
Tabla 8. Dispositivo de gama media, método 1, general.

Como se comprobará más tarde en los siguientes métodos, este dispositivo introduce un error añadido que puede ser debido a que el punto central de la imagen no coincida con el punto sin refracción.

El método 1 tiene un error global cercano al 25%. Un error que parte del 14% en distancias cortas y que va aumentando a la vez que el punto de interés se aleja.

Método 2

Distancia real de 6 metros.



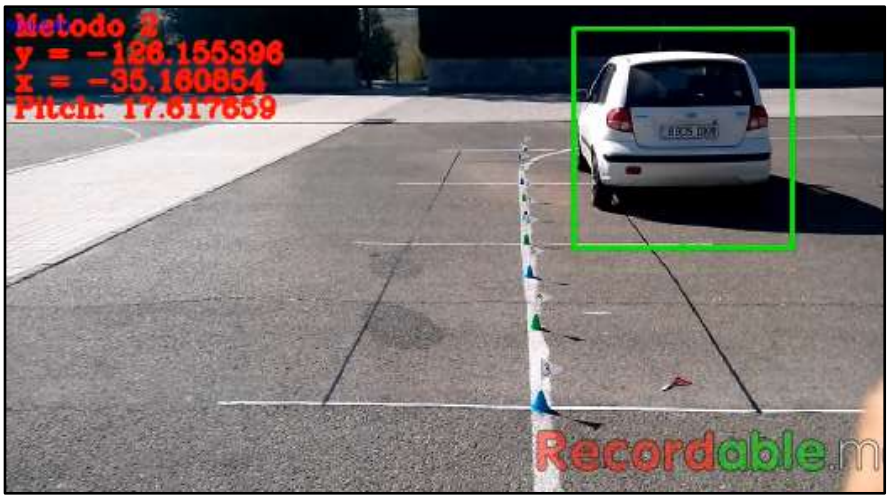
Distancia real de 9 metros.

Distancia real de 4 metros.

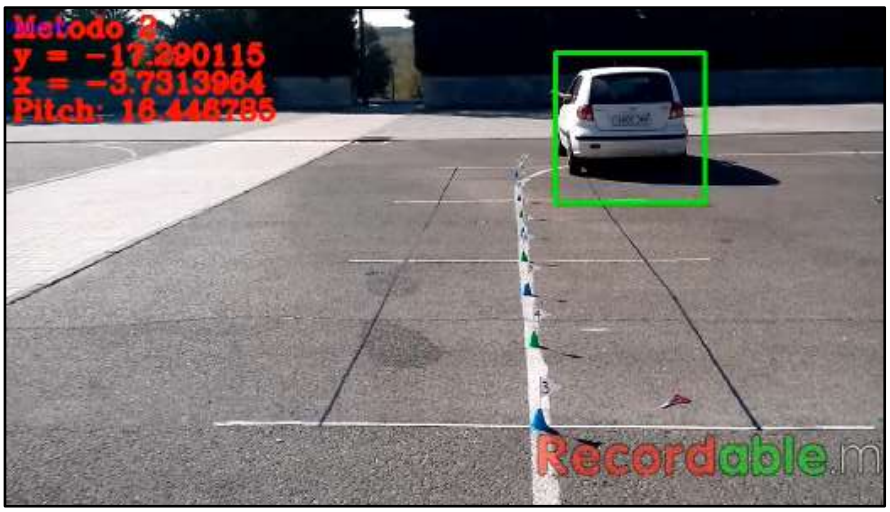
Distancia real de 6 metros.

Distancia real de 9 metros.

Distancia real de 4 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	-	-	-	-	-	31,78%
	0,37	6	4,26	1,74	29%	
	-0,03	9	5,89	3,11	34,56%	
PITCH MEDIO	7,92	4	4,77	0,77	19,25%	48,45%
	8,53	6	8,40	2,40	40%	
	8,54	9	16,75	7,75	86,11%	
PITCH ALTO	14,76	4	9,43	5,43	135,75%	876,79%
	17,62	6	-126,15	132,15	2202,5%	
	16,45	9	-17,29	26,29	292,11%	

Tabla 9. Dispositivo de gama media, método 2.

El error medio asociado a las distancias, se ha calculado sin tener en cuenta los datos referentes al pitch alto, ya que son tan disparatados que se desprecian:

	Error medio
DISTANCIA CORTA	19,25%
DISTANCIA MEDIA	34,5%
DISTANCIA LARGA	60,33%

Tabla 10. Dispositivo de gama media, método 2, general.

En este dispositivo, a diferencia del de gama baja, si toma valores coherentes con algo de pitch, aunque con unos errores muy elevados y solo con un pitch medio. El error global ronda 40%, algo que parece muy excesivo.

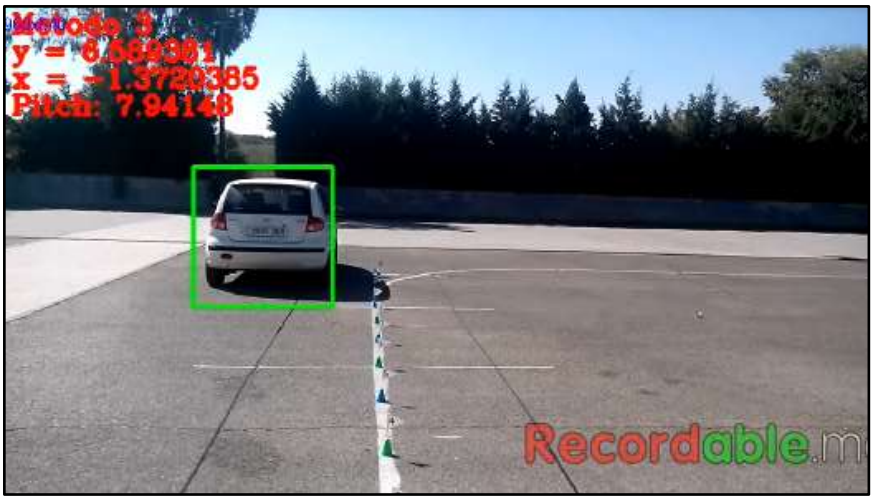
Método 3

Distancia real de 6 metros.

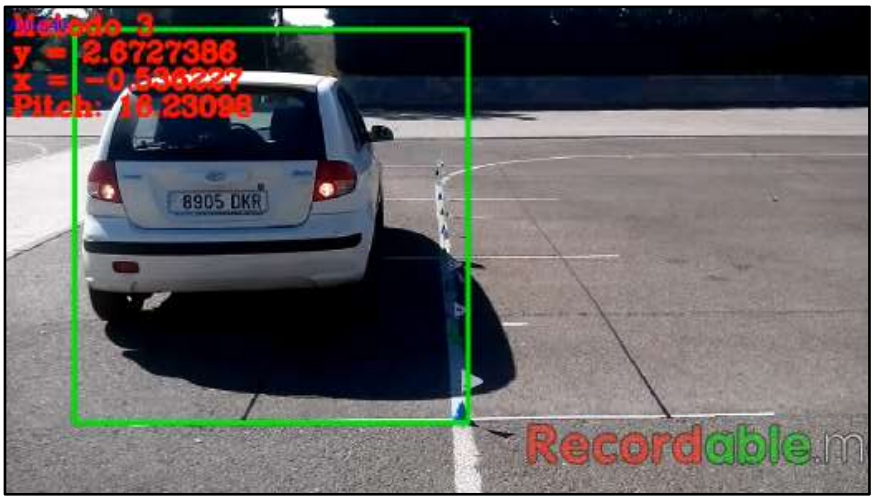
Distancia real de 9 metros.

Distancia real de 4 metros.

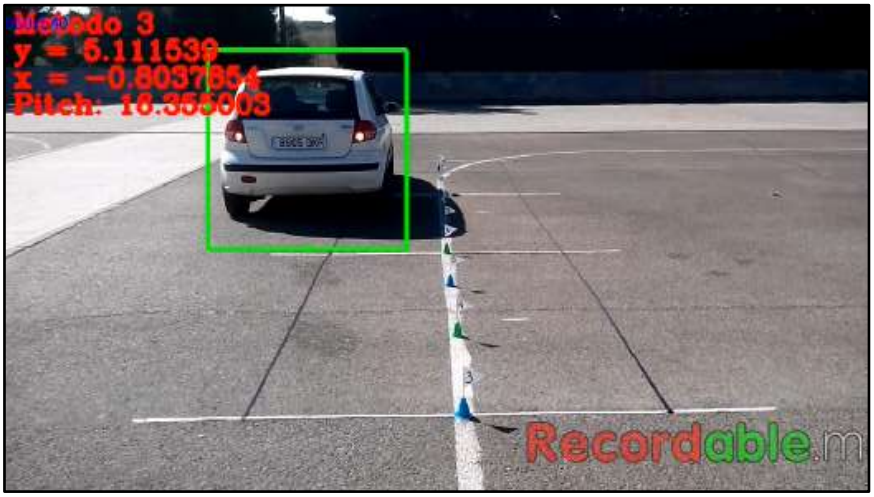
Distancia real de 6 metros.



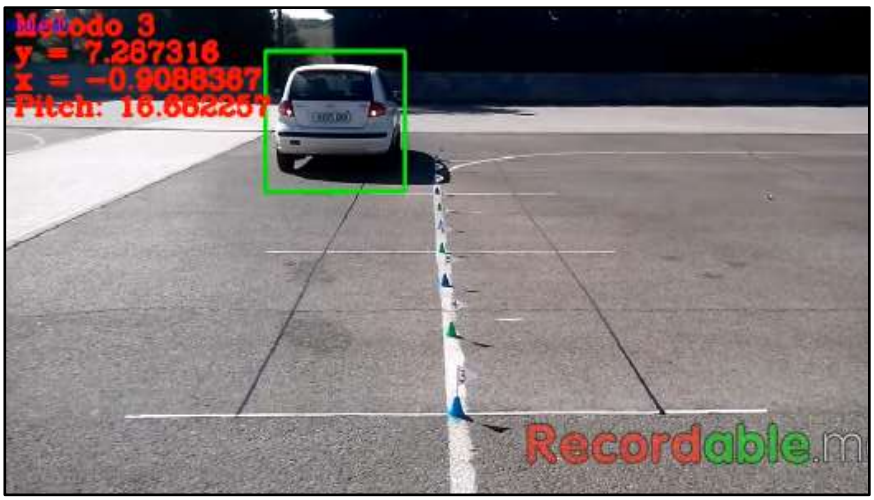
Distancia real de 9 metros.



Distancia real de 3 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	-	-	-	-	-	32,03%
	0,05	6	4,23	1,77	29,5%	
	0,01	9	5,89	3,11	34,56%	
PITCH MEDIO	8,11	4	3,35	0,65	16,25%	21,62%
	8,01	6	4,69	1,31	21,83%	
	7,94	9	6,59	2,41	26,78%	
PITCH ALTO	16,23	3	2,67	0,33	11%	14,94%
	16,35	6	5,11	0,89	14,83%	
	16,66	9	7,29	1,71	19%	

Tabla 11. Dispositivo de gama media, método 3.

	Error medio
DISTANCIA CORTA	13,62%
DISTANCIA MEDIA	22,05%
DISTANCIA LARGA	26,78%

Tabla 12. Dispositivo de gama media, método 3, general.

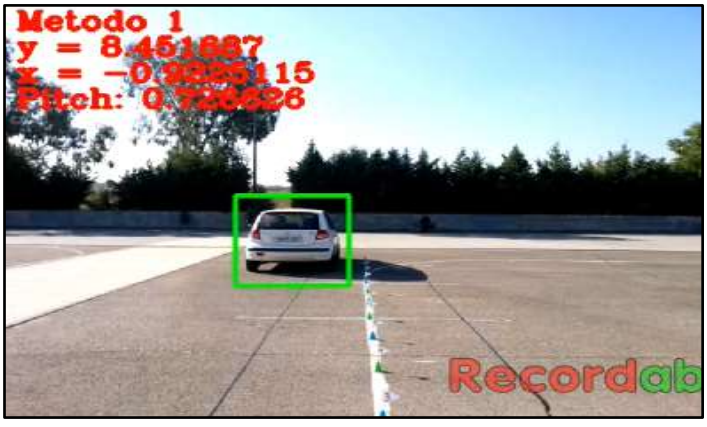
El método 3 tiene un error global de menos del 23%, muy similar al del método 1 pero ambos insuficientes para este dispositivo. Cabe destacar que este método, al igual que el resto aumenta su error al alejar el vehículo, pero además tiene mayor fiabilidad cuanto más inclinado está el dispositivo.

5.3 Samsumg Galaxy S3, dispositivo de gama alta

Método 1

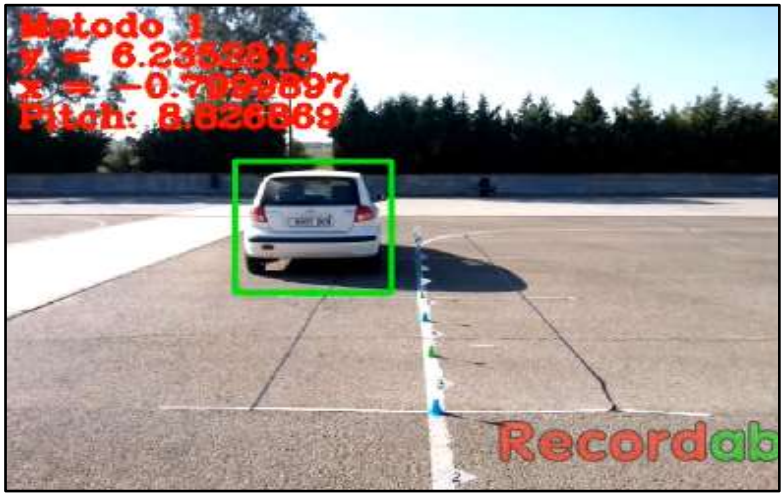
Distancia real de 3,70 metros.

Distancia real de 6 metros.

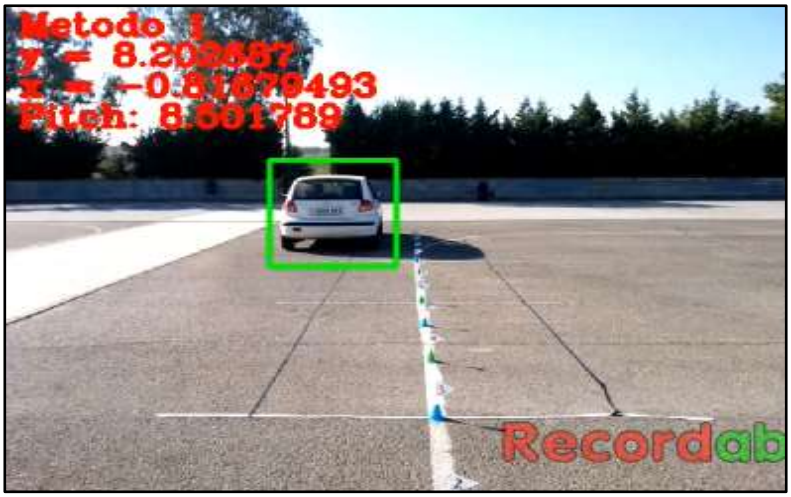


Distancia real de 9 metros.

Distancia real de 3 metros.

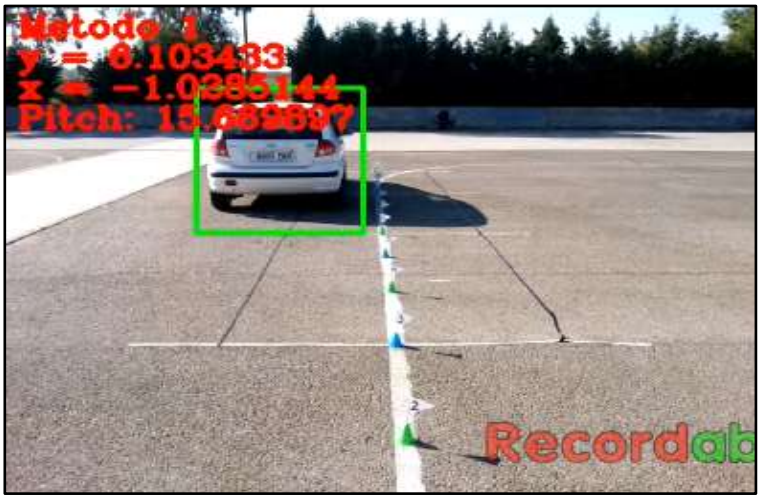


Distancia real de 6 metros.

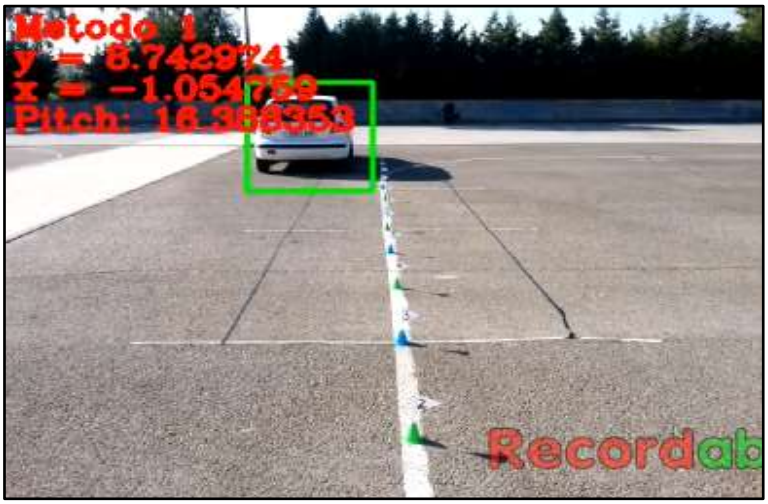


Distancia real de 9 metros.

Distancia real de 3 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	0,85	3,70	3,74	0,04	1,08%	2,95%
	0,52	6	5,90	0,10	1,67%	
	0,73	9	8,45	0,55	6,11%	
PITCH MEDIO	8,80	3	3,14	0,14	4,67%	5,8%
	8,83	6	6,23	0,23	3,83%	
	8,50	9	8,20	0,80	8,89%	
PITCH ALTO	15,71	3	3,06	0,06	2%	2,18%
	15,67	6	6,10	0,10	1,67%	
	16,39	9	8,74	0,26	2,89%	

Tabla 13. Dispositivo de gama alta, método 1.

	Error medio
DISTANCIA CORTA	2,58%
DISTANCIA MEDIA	2,39%
DISTANCIA LARGA	5,96%

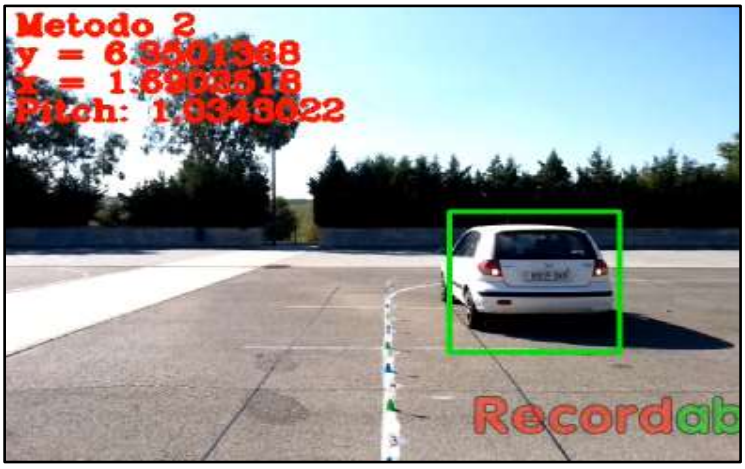
Tabla 14. Dispositivo de gama alta, método 1, general.

La primera diferencia que se aprecia con dispositivo de gama alta, que se han podido obtener todas las capturas, su ángulo de visión amplia es mayor que en el resto y al rotar el Smartphone no se pierde la información de la zona de interés.

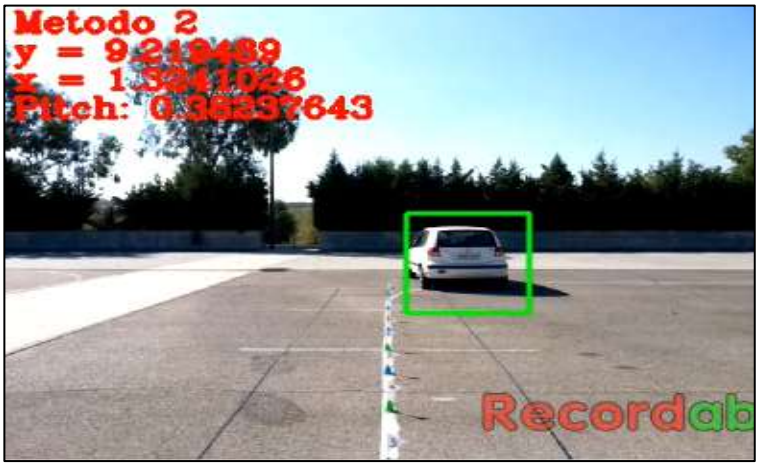
El método 1 ofrece unos resultados excelentes tanto con diferentes inclinaciones, como a distintas distancias. El error global es del 3,6% aproximadamente.

Método 2

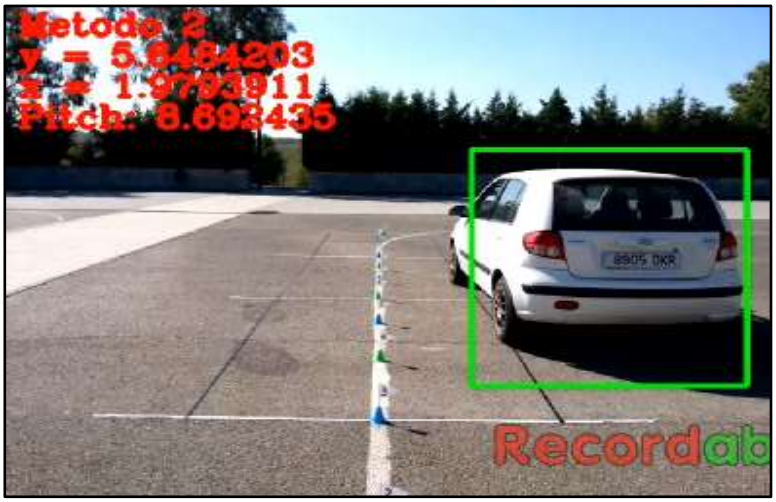
Distancia real de 4 metros.



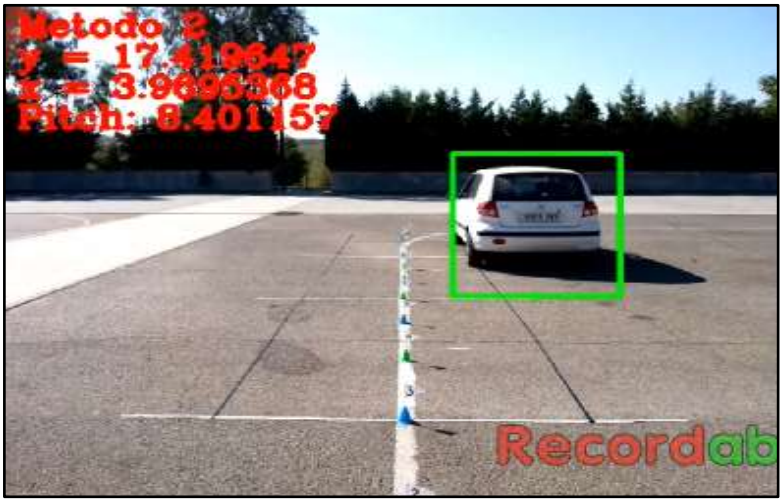
Distancia real de 6 metros.



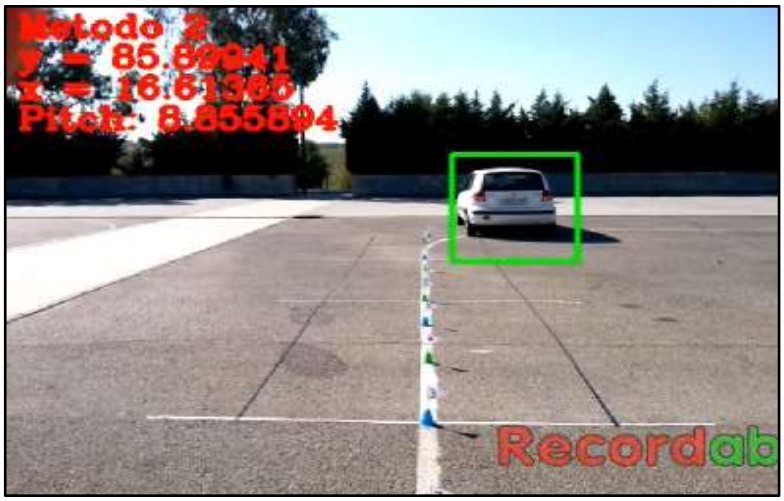
Distancia real de 9 metros.



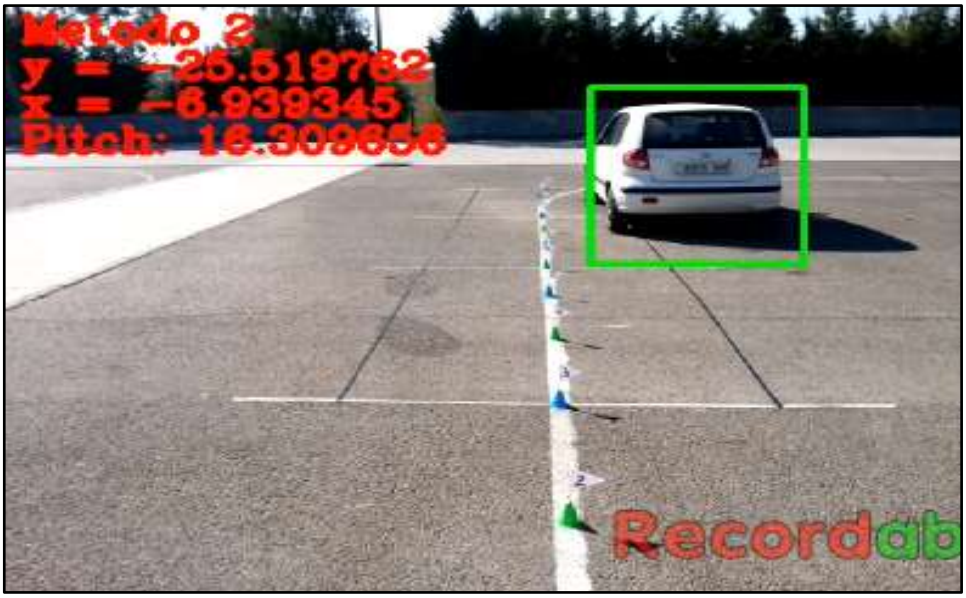
Distancia real de 3,50 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.



Distancia real de 6 metros.

Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	*0,28*	*4*	*4,10*	*0,10*	*2,5%*	*3,63%*
	1,03	*6*	*6,35*	*0,35*	*5,83%*	
	0,38	*9*	*9,23*	*0,23*	*2,55%*	
PITCH MEDIO	*8,69*	*3,50*	*5,65*	*2,15*	*61,43%*	*368,73%*
	8,40	*6*	*17,42*	*11,42*	*190,33%*	
	8,86	*9*	*85,90*	*76,9*	*854,44%*	
PITCH ALTO	*-*	*-*	*-*	*-*	*-*	*373,88%*
	16,31	*6*	*-25,52*	*31,52*	*525,33%*	
	16,61	*9*	*-11,02*	*20,02*	*222,44%*	

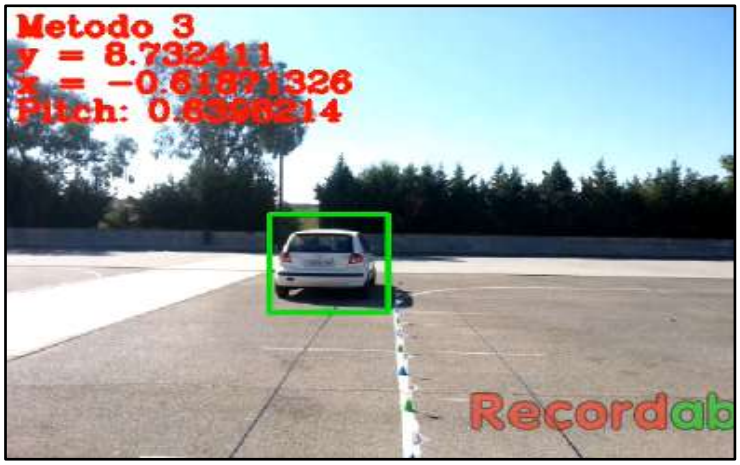
Tabla 15. Dispositivo de gama alta, método 2.

Como en los casos anteriores, el método 2 da unos resultados inadmisibles en cuanto se le aplica algo de inclinación al Smartphone. Sin pitch, el error es muy pequeño, 3,63%.

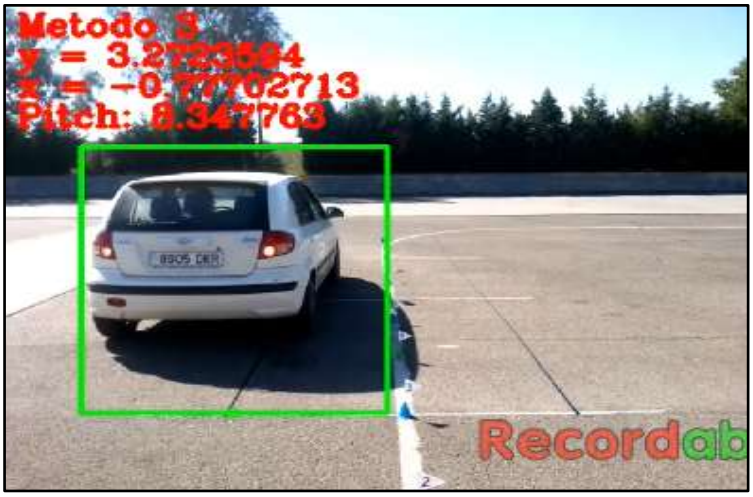
Método 3

Distancia real de 3,20 metros.

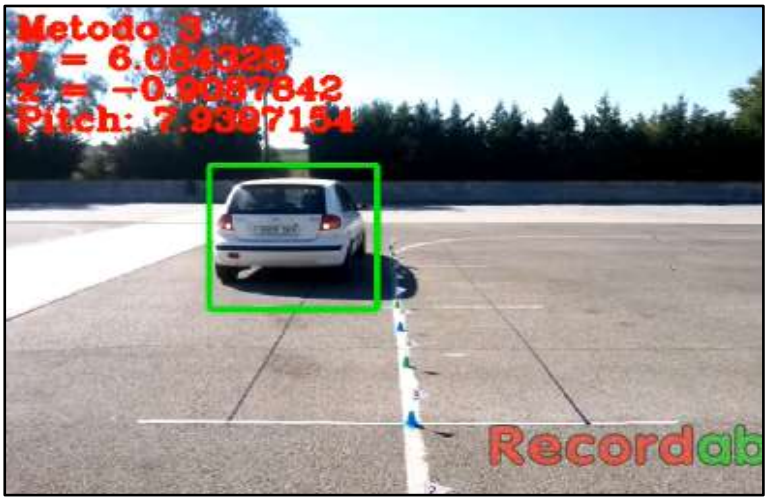
Distancia real de 6 metros.



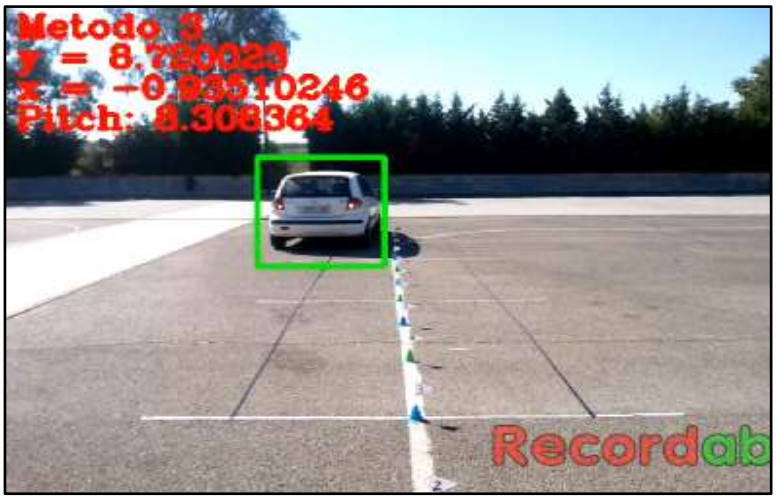
Distancia real de 9 metros.



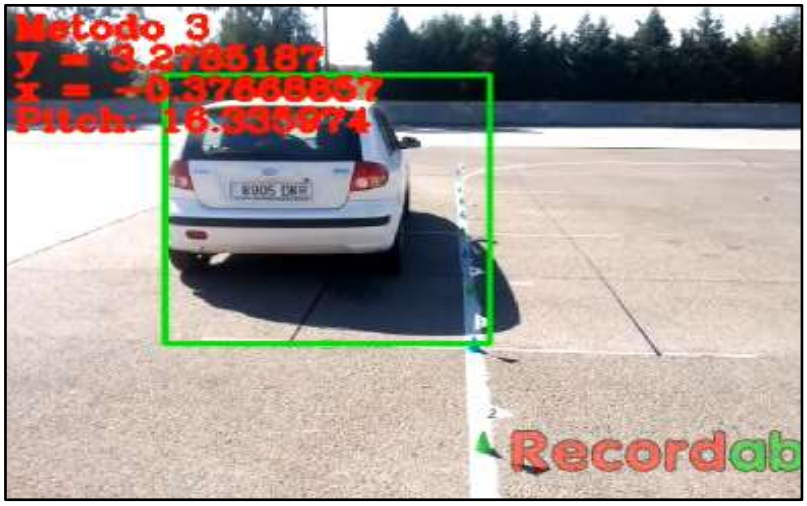
Distancia real de 3 metros.



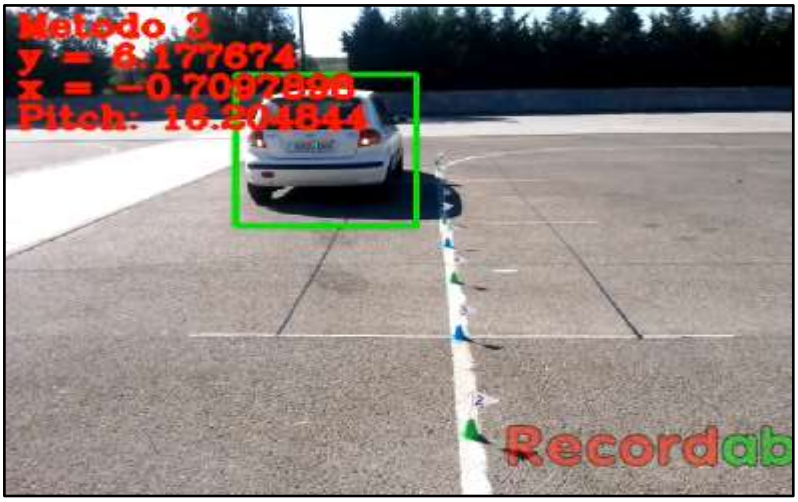
Distancia real de 6 metros.



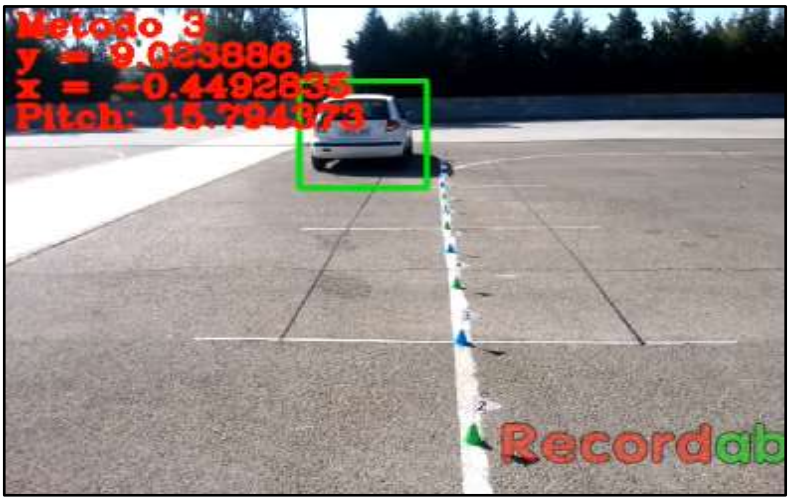
Distancia real de 8,50 metros.



Distancia real de 3 metros.



Distancia real de 6 metros.



Distancia real de 9 metros.

Los resultados numéricos se recogen en las siguientes tablas:

	Pitch real	Distancia real	Distancia estimada	Error absoluto	Error relativo	Error medio
SIN PITCH	0,53	3,20	3,46	0,26	8,12%	3,87%
	0,38	6	5,97	0,03	0,5%	
	0,64	9	8,73	0,27	3%	
PITCH MEDIO	8,35	3	3,27	0,27	9%	4,31%
	7,94	6	6,08	0,08	1,33%	
	8,30	8,50	8,72	0,22	2,59%	
PITCH ALTO	16,33	3	3,28	0,28	9,33%	4,18%
	16,20	6	6,18	0,18	3%	
	15,79	9	9,02	0,02	0,22%	

Tabla 16. Dispositivo de gama alta, método 3.

	Error relativo
DISTANCIA CORTA	8,82%
DISTANCIA MEDIA	1,61%
DISTANCIA LARGA	1,94%

Tabla 17. Dispositivo de gama alta, método 3, general.

El método 3 proporciona unos errores similares al método 1, cercanos al 4%, reflejando la buena fiabilidad de los cálculos. Llama la atención como estas pruebas dan un error superior a cortas distancias, en torno al 9%, mientras que en distancias medias y altas el error no llega ni al 2%.

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1 Conclusiones

El objetivo general del proyecto era crear una biblioteca para el cálculo de distancias; los métodos que mejor resultados han dado son el primero y el tercero. Aunque ambos métodos poseen unos resultados globales muy similares, *ver tabla 18*, el tercer método es con el que se han conseguido las distancias más precisas.

	Error gama baja	Error gama media	Error gama alta	Error global
METODO 1	*16,11%*	*25,13%*	*3,64%*	*14,96%*
METODO 3	*11,20%*	*22,86%*	*4,12%*	*12,73%*

Tabla 18. Errores globales.

Otro punto del que se han extraído conclusiones a lo largo del trabajo, es la calidad del dispositivo y cómo afecta a los resultados. En general, las tres gamas probadas tenían suficiente capacidad para asumir los cálculos sin colapsar. En cuanto a la cámara, queda reflejado, que a menor resolución de la imagen (megapíxeles) hay un mayor error a medida que el vehículo se aleja.

Finalmente, se deben tener en cuenta los logros en cuanto al trabajo personal; en este proyecto, pese a que en un principio no fue una motivación, se han adquirido los conocimientos necesarios para trabajar en un entorno de desarrollo enfocado a Android (detallado en el punto de software). Entre estos conocimientos entran nociones de java, (que hasta entonces no conocía), el lenguaje XML ("lenguaje de marcas extensible") y los conceptos necesarios para agrupar y organizar el código dejándolo preparado para que sea sencillo y útil en las aplicaciones futuras. También ha sido necesario adquirir unos conocimientos más profundos sobre sistemas de percepción pues son la base de toda la metodología de cálculo. En general, el proyecto es una buena manera de completar la formación académica, consiguiendo profundizar en cualquier tema elegido, con las directrices de un tutor, pero con la responsabilidad de un aprendizaje autónomo.

6.2 Líneas futuras

La tecnología nos ha demostrado que cualquier iniciativa se puede hacer realidad, y con el paso del tiempo mejorarse y optimizarse creando mucho más de la idea inicial. Actualmente tenemos el gran poder de la información sin barreras y solo la creatividad marca los límites.

Para la tecnología automovilística se suma el factor de que lo que hay en juego son vidas humanas, por eso es muy importante la consistencia de la aplicación, la fiabilidad numérica y las estadísticas de acierto. Además en muchas ocasiones ni superando la fiabilidad es suficiente, un ejemplo de esto es el coche de google, pues aunque es totalmente autónomo, debe colocar volante y pedales para cumplir con ley *[33].*

Otro factor que hace más complejas estas aplicaciones es la confianza de las personas, pues nos cuesta adaptarnos a las nuevas tecnologías. Pero vemos como paulatinamente nuestros coches son más autónomos y nuestros móviles cada vez tienen mayor potencial de procesamiento haciendo posible desarrollar este tipo de aplicaciones.

7. BIBLIOGRAFÍA

[1] DGT: Las principales cifras de la siniestralidad vial. España 2012. Disponible en: http://www.dgt.es/Galerias/seguridad-vial/estadisticas-e-indicadores/publicaciones/principales-cifras-siniestralidad/cifras_siniestralidad_2012.pdf

[2] OMS: Informe sobre la situación mundial de la seguridad vial, 2009. Disponible en: http://www.who.int/violence_injury_prevention/road_safety_status/2009/es/

[3] FITSA: El valor de la seguridad vial. España 2008. Disponible en: http://stopaccidentes.org/uploads/file/Costes_accidentes.pdf

[4] CEA (Comisariado Europeo del Automóvil): Seguridad activa y pasiva del vehículo. Disponible en: http://www.cea-online.es/reportajes/seguridad.asp

[5] Motor: principales causas de los accidentes de tráfico. Disponible en: http://motor.excite.es/accidentes-de-trafico-mas-comunes.html

[6] Xataka, apasionados por la tecnología: Sistemas que vigilan la salud del conductor. Disponible en: http://www.xataka.com/tecnologia-en-el-coche/tecnologia-para-el-coche-sistemas-que-vigilan-la-salud-del-conductor

[7] Xataka, apasionados por la tecnología: Sistemas de detección de coches. Disponible en: http://www.xataka.com/gadgets-y-coches/sistemas-de-deteccion-en-los-coches-para-evitar-accidentes

[8] Mercedes-Benz: Control de ángulo muerto. Disponible en: http://m.mercedes-benz.es/es_ES/blind_spot_assist/detail.html

[9] Mercedes-Benz: Reconocimiento automático de señales de tráfico. Disponible en: http//techcenter.mercedes-benz.com/es_ES/traffic_sign_assist/detail.html

[10] Ford: Control de velocidad crucero adaptativa (ACC). Disponible en: http://www.ford.com.ar/servlet/Satellite?c=DFYArticle&cid=1249063459431&pagename=FAR%2Fcontroller&site=FAR

[11] Motor Pasión Futuro: Continental ContiGuard, prevención de accidentes mediante la detección de objetos. Disponible en: http://www.motorpasionfuturo.com/ayudas-a-la-conduccion/continental-contiguard-prevencion-de-accidentes-mediante-la-deteccion-de-objetos

[12] Motor Pasión Futuro: AMULETT CAR-2-X, sistemas de radio para evitar atropellos. Disponible en: http://www.motorpasionfuturo.com/ayudas-a-la-conduccion/amulett-car-2-x-sistema-de-radio-para-evitar-atropellos

[13] Bosch: tecnología para la automoción de Bosch en España. Disponible en: http://www.xn--bosch-tecnologadelautomvil-roc1p.es/es/es/driving_safety_8/driving_safety_systems_for_commercial_vehicles_8/driver_assistance_systems_33/driver_assistance_systems_3.html

[14] GIZMO: El nuevo coche autónomo de Google. Disponible en: http://es.gizmodo.com/el-nuevo-coche-autonomo-de-google-no-tiene-volante-sol-1582478700

[15] Explicando: ¿Cómo funciona el coche sin conductor de Google? Disponible en: http://www.explicando.es/como_funciona_el_coche_sin_conductor_de_google_77

[16] Waze: Obtén la mejor ruta, todos los días, con ayuda en tiempo real de otros conductores. Disponible en: https://www.waze.com/es/

[17] Google play: iOnRoad Augmented Driving. Disponible en: https://play.google.com/store/apps/details?id=com.picitup.iOnRoad&hl=es

[18] Google play: Drivea-Driving Assistant App. Disponible en: https://play.google.com/store/apps/details?id=com.driveassist.experimental&hl=es

[19] Google play: Car Home Ultra. Disponible en: https://play.google.com/store/apps/details?id=spinninghead.carhome&hl=es

[20] Google play: Dirección General de Tráfico. Disponible en: https://play.google.com/store/apps/details?id=es.quadram.dgt

[21] El Androide Libre: Los Smartphones como sustitutos de las cámaras digitales. Disponible en: http://www.elandroidelibre.com/2013/11/los-smartphones-como-sustituto-de-las-camaras-digitales.html

[22] ITexpresso: ¿Smartphones o cámaras compactas? La guerra de los megapíxeles. Disponible en: http://www.itespresso.es/smartphones-camaras-compactas-guerra-megapixeles-121184.html

[23] How does the iPhone 4S camera stack-up against other cameras? Disponible en: http://campl.us/posts/iPhone-Camera-Comparison

[24] Omicrono: Así funciona el acelerómetro de tu Smartphone. Disponible en: http://www.omicrono.com/2012/05/asi-funciona-el-acelerometro-de-tu-smartphone/

[25] Xataka, apasionados por la tecnología: Android acapara el 80% de la cuota de mercado. Disponible en: http://www.xataka.com/moviles/android-ya-acapara-el-80-de-cuota-de-mercado-en-smartphones-y-la-mitad-son-telefonos-samsung

[26] Gowex, Ideup: Desarrollador Android vs iOs. Disponible en: http://www.ideup.com/blog/desarrollador-android-vs-desarrollador-ios

[27] Android Developers: CameraParameters. Disponible en: http://developer.android.com/reference/android/hardware/Camera.Parameters.html

[28] Android Developers: Sensor. Disponible en: http://developer.android.com/reference/android/hardware/Sensor.html

[29] Android Developers: SensorManager. Disponible en: http://developer.android.com/reference/android/hardware/SensorManager.html

[30] Android Developers: SensorEventListener. Disponible en: http://developer.android.com/reference/android/hardware/SensorEventListener.html

[31] Android Developers: SensorEvent. Disponible en: http://developer.android.com/reference/android/hardware/SensorEvent.html

[32] DE LA ESCALERA HUESO, Arturo. Visión por Computador, fundamentos y métodos. Isabel Capella; Sonia Ayerra; José A. Clares; Tini Cardoso. Pearson Educación, S.A. Madrid, 2001. ISBN: 84-205-3098-0

[33] They Are News, Ciencia y Tecnología: La ley obliga a Google a poner volante y pedales a su vehículo autónomo. Disponible en: http://www.theyarenews.com/2014/8/22/obliga-google-poner-volante-pedales-vehiculo-autonomo-19495.asp